SH20-1620-1

Installed User Program

APL Econometric Planning Language for System/370 OS/VS1 or OS/VS2 or VM/370 CMS Program Description/Operations Manual

Program Number: 5796-PDW

This manual describes the functional capabilities of the APL Econometric Planning Language program which is designed as an interactive APL based language for econometric modeling and forecasting. The program provides features for dealing with economic variables (primarily time-series), such as a set of functional and economic operators, tabular and graphic display, parameter estimation, model solution and file building. APLSV, PRPQ #WE 1191, Program Number 5799-AJF; or APL/CMS, PRPQ #MFZ2608, Program Number 5799-ALK; or VS APL, Program Number 5748-AP1 is required for installation of this program.

This manual is also an installation and operations reference document.

IBM

PROGRAMMING SERVICES PERIOD

During a specified number of months immediately following initial availability of each licensed program, designated as the PROGRAMMING SERVICES PERIOD, the customer may submit documentation to a designated IBM location when he encounters a problem which his diagnosis indicates is caused by a licensed program error. During this period only, IBM through the program sponsor(s), will, without additional charge, respond to an error in the current unaltered release of the licensed program by issuing known error correction information to the customer reporting the problem and/or issuing corrected or notice of availability of corrected code. However, IBM does not guarantee service results or represent or warrant that all errors will be corrected. Any onsite programming services or assistance will be provided at a charge.

WARRANTY

EACH LICENSED PROGRAM IS DISTRIBUTED ON AN 'AS IS' BASIS WITHOUT WARRANTY OF ANY KIND EITHER EXPRESS OR IMPLIED.

Second Edition (June 1976)

A form for readers' comments is provided at the back of this publication. If the form has been removed, comments may be addressed to IBM Corporation, B/O 464, 1801 K. Street, Washington, D.C. 20006. Attention: Andy J. Haire.

APL ECONOMETRIC PLANNING LANGUAGE

Program Description/Operations Manual

TABLE OF CONTENTS:

I INTRODUCTION

THE APL ECONOMETRIC PLANNING LANGUAGE is an interactive planning language based upon APL. It provides features for dealing with economic variables (primarily time-series), such as data analysis and transformation, tabular and graphic display, parameter estimation, model solution and file handling.

The APL ECONOMETRIC PLANNING LANGUAGE provides the practicing economist, or the teacher, with easy-to-use tools for interactive model building and solving. Care has been taken to use much of the power inherent in APL, such as its rich set of primitive functions and its compact data handling and logic capabilities. Operators on time series have been designed so as to be compatible with the underlying language.

As a consequence, the APL ECONOMETRIC PLANNING LANGUAGE constitutes a truly interpretive and interactive system, with all attendant advantages. The objects with which it deals are APL functions and variables. Therefore, the user can easily add his own functions to tailor the language to his specific requirements.

For heavy usage, data file routines have been added, based upon the processor TSIO of APLSV or AP110 of APL/CMS. The user can set up and utilize his own data base by means of simple store, retrieve and update functions. Those users who already have data bases, should find little difficulty in adding their own interface functions via TSIO or AP110.

The system was designed and developed at the IBM Philadelphia Scientific Center (see refs. 10-12), with consultation from IBM's Business Planning Department, Stuttgart, and Economic Research Department, Armonk. It is installed at the IBM Economic Research Department, Armonk, and the APL Design Group, Palo Alto.

The user of the APL ECONOMETRIC PLANNING LANGUAGE may wish to become somewhat familiar with APL itself. Relevant documentation on APL is available in the APL/360 Primer (GH20-0689), the APL/360 User's Manual (GH20-0906), and the APL Language Manual (GC26-3847).

II SYSTEM OVERVIEW

The system comes in the form of two APL workspaces on one tape: EPL1, EPL2. It was developed and can be used in one 80K byte workspace, but has been regrouped into two workspaces of size less than 65K.

The functions of the system are partitioned into seven groups.

EPL1 contains:

BASEGRP
DISPLAYGRP
PLOTGRP

EPL2 contains:

FILEGRP - for use with APLSV
FILEGRPC - for use with APL/CMS
REGRESSGRP
SOLVEGRP
TRANSLATEGRP
ADDENDUM

The user will work with an active workspace which will contain all those groups which he needs at a particular time. The active workspace must always contain *BASEGRP*. The other groups are "copied" (see III OPERATIONS) as needed. They contain those functions which are available to the user:

DISPLAYGRP:	*DISPLAY TABULATE*
PLOTGRP:	*PLOT PLOTF VS*
FILEGRP:	*CDATA DDATA EDATA RDATA RETRIEVE STORE*
FILEGRPC:	*CDATAC DDATAC EDATAC RDATAC RETRIEVEC STOREC*
REGRESSGRP:	*A1 COR DIST GLS INST PCINST REGRESS*
SOLVEGRP:	*ORDER SOLVE SOLVECON SOLVEIT SOLVEREL SSOLVE SSOLVEF STRUCTURE*
TRANSLATEGRP:	*DO TRANSLATE*
ADDENDUM:	*RETRANS VOCABULARY UPDATE PREP TABULTEXT*

In addition, these groups may contain auxiliary functions which can not be called directly by the user and are not described in this report.

The user will normally work on a 2741 terminal with APL type ball 987, and will have no difficulty in composing function names with understruck capital letters. All functions listed above are described in this manual to an extent sufficient for their use. If the user at some point wants to make changes and additions to the system, he may wish to consult the SYSTEMS GUIDE.

In the systems guide all functions are listed , most of them alphabetically. The alphabetic order is not observed for the functions described in the ADDENDUM of the PDOM (the functions in the group ADDENDUM of EPL2). They were added as an afterthought and are listed together at the end of the systems guide.

In a sense this is not undesirable. The addition of these functions provides the user with an example of the extensibility of the system.

IIa PROGRAM INSTALLATION

For use under APLSV, the APL Econometric Planning Language workspaces are loaded into the APLSV public library using the system-supplied utility. Under VM/370 CMS, the program's APLSV workspaces can be converted to APL/CMS workspaces and loaded into the VM/370 system, using the system-supplied utility. For use under VS APL, the APL/CMS workspaces are converted to VSAPL using the system-supplied utility.

The file-handling group *FILEGRP* will operate under APLSV; the file-handling group *FILEGRPC* will operate under CMS, in either the APL/CMS version or the VS APL version running under VM/370 CMS, but not in VS APL under a VSPC environment. Except for file-handling, the program will work in VSAPL under a VSPC environment.

III OPERATION

The system has been designed such that very little knowledge of APL is required for its operation. The user who has no familiarity with APL can start a terminal session in the following fashion:

1. Load the workspace EPL1 by the command:)*LOAD EPL*1

2. Change the (active) workspace name to, say, TEST:

)*WSID TEST*.

3. Work in the active workspace TEST by following the examples in this manual. If desirable, save the active workspace by the command:)*SAVE*.

4. "copy" other functions from EPL2 as required, e.g.:

)*COPY EPL*2 *REGRESSGRP* to obtain all functions of the group REGRESSGRP, or

)*COPY EPL*2 *INST* to obtain only the function *INST* of the REGRESSGRP.

   Groups or functions (or variables) no longer needed can be removed from the active workspace by commands such as:)*ERASE DISPLAYGRP* or)*ERASE TABULATE*.

IV SYSTEM DESCRIPTION

This manual describes the APL ECONOMETRIC PLANNING LANGUAGE. It is assumed that the reader is familiar with the principles of econometrics, but references to the relevant literature will be given in the text. The APL ECONOMETRIC PLANNING LANGUAGE concentrates on parameter estimation for multivariate models and on numerical solution of these models. According to the usual steps in practical econometric work the report is divided into the chapters

TIME SERIES: DEFINITION AND DISPLAY

MODEL LANGUAGE

REGRESSION

MODEL SOLUTION

DATA FILE ROUTINES

To demonstrate the capabilities of the APL ECONOMETRIC PLANNING LANGUAGE an example is repeatedly used. Context and data are taken from BASS [2], but the formulation of the model is changed. The model describes the interrelation between sales and advertising expenditures in the US cigarette industry.

SAMPLE MODEL

(1) $S1(t) = a(1,1)W1(t)/W2(t) + a(1,2)S1(t-1) + a(1,3)DI(t) + u(1,t)$

(2) $S2(t) = a(2,1)W1(t)/W2(t) + a(2,2)S2(t-1) + a(2,3)DI(t) + u(2,t)$

(3) $W1(t) = a(3,0) + a(3,1)S1(t)/S2(t) + a(3,2)S1(t-1)/S2(t-1) + u(3,t)$

(4) $W2(t) = a(4,0) + a(4,1)S1(t)/S2(t) + a(4,2)S1(t-1)/S2(t-1) + u(4,t)$

S1(t), [S2(t)] = sales per capita of filter [nonfilter] cigarettes
= years 1953 to 1965
W1(t), [W2(t)] = advertising expenditures per capita for filter [nonfilter] cigarettes ($)

DI(t) = disposable personal income per capita ($)
u(n,t) = disturbances.

W1(t), W2(t), DI(t) are deflated by appropriate price indices. The population comprises all persons over 20 years of age.

TABLE 1: Data for the sample model (BASS).

Year	S1	S2	W1	W2	DI
1953	250.328	3196.318	.055	.520	2609.336
1954	398.658	2856.406	.126	.425	2622.769
1955	690.081	2668.824	.236	.371	2785.544
1956	952.423	2453.522	.367	.360	2893.810
1957	1232.111	2178.211	.453	.279	2914.137
1958	1414.718	2008.075	.451	.289	2904.290
1959	1526.195	2006.827	.495	.287	3018.978
1960	1571.303	1991.590	.462	.292	3055.062
1961	1613.578	1971.696	.455	.286	3115.875
1962	1657.487	1900.684	.434	.283	3221.365
1963	1731.530	1808.007	.518	.285	3298.679
1964	1672.207	1590.523	.506	.283	3472.402
1965	1702.119	1633.202	.486	.317	3658.895

The technical part of econometric modeling consists of 1. the statistical estimation of the unknown values for the parameters a(1,1)...a(4,2) and 2. the solution of the model for the <u>endogenous</u> variables S1(t), S2(t), W1(t) and W2(t), when values for the parameters and the <u>exogenous</u> variables DI(t), S1(t-1) and S2(t-1) are provided. A solution can be computed either for historical values of the exogenous variables (model validation) or for estimated future values (forecasting).

The system provides <u>limited information</u> techniques for the estimation of models that are linear in the parameters, i.e. ordinary regression, generalized regression, instrumental variables substitution with or without principal components transformation, and polynomial distributed lags. Models may be nonlinear in their variables, as is the sample model. Little emphasis is put on the analysis of the internal structure of a single time series. The system is modular so that either the user's own APL functions or public library functions for time series analysis can be added easily.

1. TIME SERIES: DEFINITION AND DISPLAY

In a general sense, data in econometric modeling may be time series, cross-sectional observations, equations or complete models. In this chapter we will consider the definition of time series (and cross-sectional series) and various display routines. From an APL point of view time series will be represented as numeric arrays, while equations and models will have the form of character arrays; see chapter 2. Optional names can be assigned to each data object. <u>These names should not contain understruck characters.</u>

1.1. DEFINITION OF A TIME SERIES

In discrete econometric modeling <u>equidistant</u> time series are used almost exclusively. Therefore, it is sufficient to concentrate on this type of series. A <u>time series</u> with an optional *NAME* is defined by the dyadic function <u>*DF*</u>

NAME←*HDR* <u>*DF*</u> *VAL*

The right argument *VAL* contains the values of the different periods, ordered with increasing time. *VAL* is usually a vector, denoting a series with exactly one value for each period. But *VAL* can also be a higher rank input, denoting a series with multiple classification. The last coordinate must always relate to time . As an example consider a compound series *SALES*, where the first coordinate relates to different products, the second to geographically distinguished markets and the third to different sales periods.

The time characteristic of a series is described by the left argument *HDR*, a three-element vector, subsequently called <u>header</u>. *HDR*[1] is the <u>periodicity</u> of the series and is defined as the number of periods corresponding to one full year. The APL ECONOMETRIC PLANNING LANGUAGE recognizes the following valid periodicities:

1 = yearly data
2 = semi-yearly data
4 = quarterly data
12 = monthly data
52 = weekly data.

The other two elements of the left argument *HDR* determine the origin of the series, which is the first period of the *VAL* array. *HDR*[2] is the year and *HDR*[3] the period within the year. The following are valid examples of headers:

4 1970 2	quarterly series starting 2. Q. 1970
12 1970 5	monthly series starting May 1970
1 1971 1	yearly series yearly starting 1971

An error message *DF SPEC ERR* is returned if the header is not correctly specified.

The explicit result *NAME* is a global APL-variable and can be manipulated outside the APL ECONOMETRIC PLANNING LANGUAGE by any APL-primitive or function. *NAME* is basically the catenation of the header and the value array. For internal system convenience the header is decoded into a scalar. If needed, the user can encode it again by applying the function *EC*; see example below.

The function

LIST ASSIGN X

assigns names to the rows of a rank two series *X* (matrix), so that an aggregate series can be decomposed into several rank one series (vectors). *LIST* contains those names. An *ASSIGN SPEC ERR* message is returned if the number of rows in *X* does not correspond with the number of names in *LIST*. After the execution of *ASSIGN*, the various rows can be called and manipulated individually by their new names.

Example 1

1.2. CROSS-SECTIONAL SERIES

For dynamic econometric models, repetitivity of economic phenomena is associated with the time component. There are, however, also time invariant models with observation data sampled in a cross-sectional manner. This data type can also be defined by means of the function *DF* with a scalar zero header *HDR*←0. In the following we will omit the treatment of cross-sectional series, since the main focus of the APL ECONOMETRIC PLANNING LANGUAGE is on dynamic modeling. But most of the functions work also for cross-sectional data (where this is meaningful).

EXAMPLE 1

```
      COST←12 1973 5 DF 4.1 5.2 5.9 6.3 7.1 6.8 7 5.8 5.1

      COST
2368112 4.1 5.2 5.9 6.3 7.1 6.8 7 5.8 5.1

      EC COST[1]
12 1973 5

      SALES←4 1972 3 DF 2 2 3ρ2 4 3 7 5 6 3 2 11 9 13 8

      SALES
789104        2        4        3
789104        7        5        6

789104        3        2       11
789104        9       13        8

      TEST←1 1965 1 DF 3 4ρ7 3 4 9 11 9 5 7 3 17 15 13

      TEST
196601        7        3        4        9
196601       11        9        5        7
196601        3       17       15       13

      'TEST1,TEST2,TEST3' ASSIGN TEST

      TEST1
196601 7 3 4 9

      TEST2
196601 11 9 5 7
```

1.3. TABULAR DISPLAY OF TIME SERIES

The monadic function

DISPLAY LIST

displays all time series contained in the right argument *LIST*. As with all subsequently defined functions which accept a *LIST* of series as arguments, these series must be of rank one (vectors). The only exceptions are the functions in chapter 5, DATA FILE ROUTINES.

Example 2

A simultaneous tabular display of several time series is provided by the function

CTR TABULATE LIST

The right argument *LIST* is a list of rank one series, which must all have the same periodicity. The tabulation has as domain those periods common to all series (intersection of the series; for explicit control of the intersection process by the user, see chapter 1.5.). Blank lines between successive series are inserted by additional commas in *LIST*; see example.

An *INTERSECT PER ERR* message is given if periodicities do not match. No output is returned if the intersection of the series is empty.

The left argument *CTR* is a 4-component vector of parameters: *CTR*[1] is an indent in front of the first listed period. It should be at least as large as the longest name in *LIST*. *CTR*[2] defines the number of periods within a table page. Set *CTR*[2]←0 if the number of periods per page should be calculated automatically. *TABULATE* then assigns as many periods to one page as will fit according to the maximally permitted printing width (⎕*PW*). *CTR*[3] and *CTR*[4] control the format of the tabulated values. They should be chosen so that all significant positions will be printed. *CTR*[3] is the total number of positions, *CTR*[4] the number of positions after the decimal point. If the absolute value of a table entry is too large, a blank is printed instead.

A *TABULATE SPEC ERR* message is returned if *CTR* is not correctly specified.

EXAMPLE 2

DISPLAY 'COST,S1'

VARIABLE COST

PERIODICITY = 12
ORIGIN = 1973 5
NO OF ENTRIES = 9

TIME		VALUE	TIME		VALUE
1973	5	4.10000	1973	10	6.80000
1973	6	5.20000	1973	11	7.00000
1973	7	5.90000	1973	12	5.80000
1973	8	6.30000	1974	1	5.10000
1973	9	7.10000			

VARIABLE S1

PERIODICITY = 1
ORIGIN = 1953 1
NO OF ENTRIES = 13

TIME		VALUE	TIME		VALUE
1953	1	250.32800	1960	1	1571.30300
1954	1	398.65800	1961	1	1613.57800
1955	1	690.08100	1962	1	1657.48700
1956	1	952.42300	1963	1	1731.53000
1957	1	1232.11100	1964	1	1672.20700
1958	1	1414.71800	1965	1	1702.11900
1959	1	1526.19500			

EXAMPLE 3

```
10 5 9 3 TABULATE 'S1,S2,W1,W2,,DI'
```

	1953	1954	1955	1956	1957
*S*1	250.328	398.658	690.081	952.423	1232.111
*S*2	3196.318	2856.406	2668.824	2453.522	2178.211
*W*1	.055	.126	.236	.367	.453
*W*2	.520	.425	.371	.360	.279
DI	2609.336	2622.769	2785.544	2893.810	2914.137

	1958	1959	1960	1961	1962
*S*1	1414.718	1526.195	1571.303	1613.578	1657.487
*S*2	2008.075	2006.827	1991.590	1971.696	1900.684
*W*1	.451	.495	.462	.455	.434
*W*2	.289	.287	.292	.286	.283
DI	2904.290	3018.978	3055.062	3115.875	3221.365

	1963	1964	1965
*S*1	1731.530	1672.207	1702.119
*S*2	1808.007	1590.523	1633.202
*W*1	.518	.506	.486
*W*2	.285	.283	.317
DI	3298.679	3472.402	3658.895

1.4. PLOTS OF TIME SERIES

The plot functions incorporate several functions from <u>Graphs and Histograms in APL</u> [5].

The function

CTR PLOT LIST

plots all series contained in *LIST* within the same graph. The abscissa is time. If instead of time, a time series *X* is to be used as abscissa, it must be indicated by the use of *VS*:

CTR PLOT LIST VS 'X'

An *INTERSECT PER ERR* message is produced if the periodicities of the series in *LIST* do not match. Also, *X* must match if *VS* is applied. Again as in the previous chapter only the common intersection of the series is plotted.

The three element vector *CTR* controls the size and location of the graph. *CTR*[1] is the length of the ordinate (in printed vertical lines), *CTR*[2] the length of the abscissa (in printed horizontal positions) and *CTR*[3] an indent to the ordinate. If the user wants to produce a finer plot with typing element #1167114, the function *PLOTF* should be applied instead of *PLOT*. However, in this case *LIST* can contain a maximum of only two series. A *PLOTF SPEC ERR* message is returned if *LIST* refers to more than two series. Note also that the fine plot can be used only on a 2741 with correspondence keyboard.

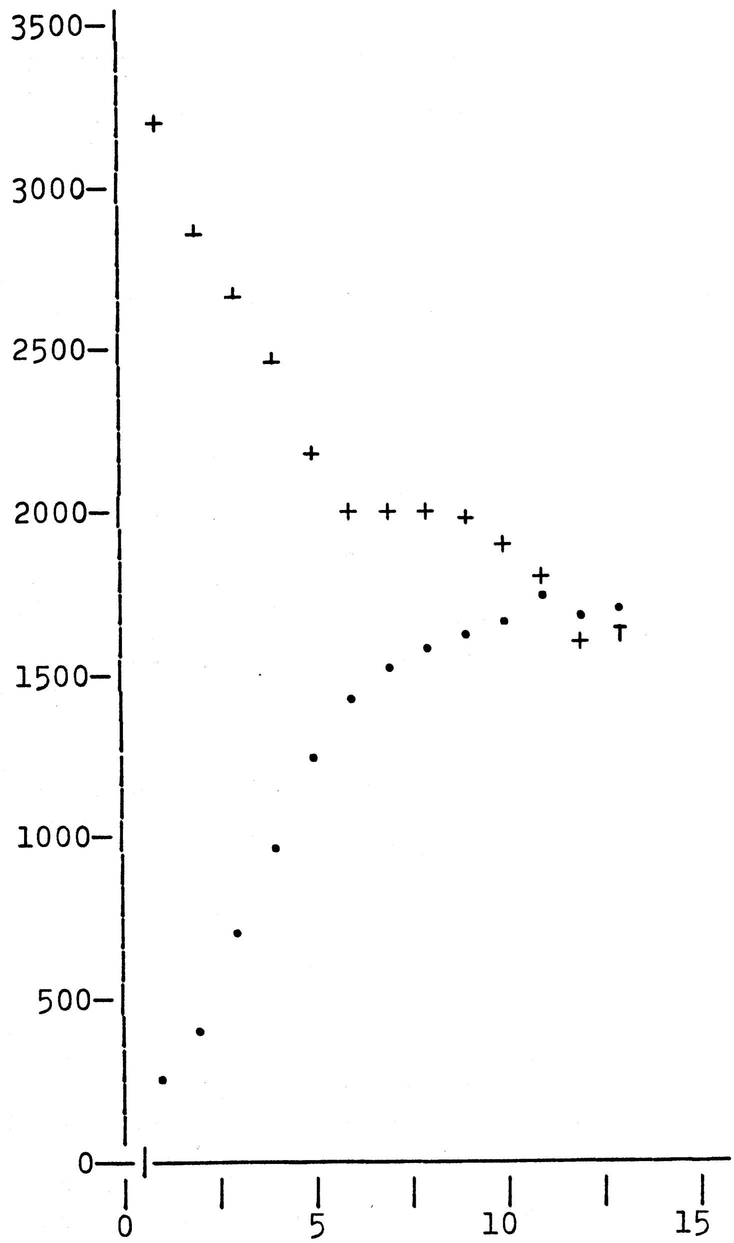
<u>Example 4</u>

1.5. EXPLICIT INTERSECTION CONTROL

In the above examples of *TABULATE* and *PLOTF* the result extended over <u>all</u> common peridos of the specified list of time series (largest common intersection). The APL ECONOMETRIC PLANNING LANGUAGE provides the user with an explicit control capability for the intersection process. He can specify the <u>time frame</u> to which the intersection process is to be restricted. However, if the specified time frame includes the largest common

EXAMPLE 4

50 40 10 *PLOTF* '*S*1,*S*2'



ABSCISSA = TIME STARTING FROM 1953
. = S1
+ = S2

intersection, the latter is taken automatically. The time frame capability allows the user to limit his analysis to a selected subset of the available data. The advantage of such a feature will become obvious in the later sections REGRESSION and MODEL SOLUTION.

A time frame is set with the dyadic function *ST*

*HDR*1 *ST* *HDR*2

*HDR*1 and *HDR*2 are the first and last periods (inclusively) of the frame. The arguments are three- element headers, as described in chapter 1.1. A specification *HDR*1←0 sets no explicit lower bound, but imposes that this bound be defined by the largest common intersectiom. Similarly, *HDR*2←0 sets no explicit upper bound. The headers can be specified in any periodicity independent of the periodicity of the series under consideration. Internally equivalent bounds are computed for all possible periodicities.

Once set, a time frame remains active for all subsequent operations. It can be erased by

0 *ST* 0

From then on the largest common intersection is taken until a new frame is specified. A *DF SPEC ERR* message occurs if one of the headers *HDR*1 or *HDR*2 is misspelled.

The current frame setting can be queried by execution of the niladic function

FRAME

It returns the lower and upper bounds for each periodicity, or a zero if no frame has been set.

The intersection of two time series *X* and *Y* is again a (possibly shorter) time series. It can be found with the function

(*NAME*←) *X* *IS* *Y*

X and *Y* must have the same rank and dimension with the exception of the last coordinate (time). The rank of the result is increased by one.

The intersection of a *LIST* of rank one series is obtained by

(*NAME←*) *INTERSECT LIST*

Both *IS* and *INTERSECT* return an empty vector if an error is detected or the intersection is empty. In the latter case *IS* causes the message *IS EMPTY ERR*. *IS* delivers an *IS RANK ERR* if *X* and *Y* do not have the same rank. An *INTERSECT SPEC ERR* for *INTERSECT* indicates that *LIST* contains no non-empty series. For both *IS* and *INTERSECT* an *INTERSECT PER ERR* is produced if the periodicities of the series do not match.

Example 5

EXAMPLE 5

1 1958 1 *ST* 1 1961 1

10 0 9 3 *TABULATE* '*S*1,*DI*'

	1958	1959	1960	1961
*S*1	1414.718	1526.195	1571.303	1613.578
DI	2904.290	3018.978	3055.062	3115.875

0 *ST* 12 1956 4

10 0 9 3 *TABULATE* '*S*1,*DI*'

	1953	1954	1955
*S*1	250.328	398.658	690.081
DI	2609.336	2622.769	2785.544

FRAME

0 0 0 1 1955 1
0 0 0 2 1955 2
0 0 0 4 1956 1
0 0 0 12 1956 4
0 0 0 52 1956 17

0 *ST* 0

FRAME

0

```
      1 1963 1 ST 0


      S←S1 IS S2

      S
196401               1731.53          1672.207         1702.119
196401               1808.007         1590.523         1633.202


      DATA←INTERSECT 'S1,S2,W1,W2,DI'

      DATA
1.9640100E5     1.7315300E3     1.6722070E3     1.7021190E3
1.9640100E5     1.8080070E3     1.5905230E3     1.6332020E3
1.9640100E5     5.1800000E¯1    5.0600000E¯1    4.8600000E¯1
1.9640100E5     2.8500000E¯1    2.8300000E¯1    3.1700000E¯1
1.9640100E5     3.2986790E3     3.4724020E3     3.6588950E3
```

2. MODEL LANGUAGE

The APL ECONOMETRIC PLANNING LANGUAGE has several arithmetic and logical operators which in effect make most of the APL primitive operators available for the manipulation of time series, i.e. arrays associated with a header. It also has some special operators frequently needed in econometric modeling. This permits the user to formulate compound expressions and models. Due to the special structure of the operands, APL operator symbols cannot be used directly but must be replaced by mnemonic functions such as *P* for + or *M* for -, etc. However, for the user who prefers symbols, a translation routine is provided to generate the mnemonic function names before further execution; see 2.5.

Two situations can be distinguished. Either an operation on one or several time series is carried out immediately in order to define a new series (e.g., taking the logarithm of a raw series and subsequently using only this transformed series), or a model is to be formulated for repeated execution. In both cases the same operators are used, but in the latter case the operations have to be stored. In the following paragraphs 2.1.,2.2. and 2.3. we shall discuss the definition and the immediate execution of operations, in 2.4. we shall treat the formulation of models. The execution of models will be partly covered in 2.5. (namely for a specific single equation case), but more extensively in chapter 4.

2.1. BASIC OPERATORS ON TIME SERIES

A dyadic operator with a symbolic name *F*

(*NAME*←) *X* *F* *Y*

returns the following result:

a) *X* and *Y* are both time series. The operation *F* will be performed for the last coordinate (time) of the intersection *X* *IS* *Y* (see 1.5.). A possible setting of a time frame is observed. An optional *NAME* can be assigned to the result to define a new series, or the result can be immediately used as input to a subsequent operation; see examples below. Periodicity and rank of *X* and *Y* must be equal, also the dimension vector except for the last coordinate. Otherwise intersection error messages will be printed and an empty vector result returned; see 1.5.

b) Either X or Y is a time series and the other argument a scalar. The operation takes place between all values of the series and the scalar. A time frame setting is <u>not</u> observed.

c) Both X and Y are scalars. The operation is performed between the two scalars in the usual way. This "degenerate case" might be replaced by the direct use of the APL primitive symbol.

A <u>monadic operator</u> with the symbolic name $\underline{H}$

($NAME\leftarrow$) $\underline{H}$ X

accepts either a time series or a scalar argument X. In the first case the operation is executed for all values of the series. A time frame setting is <u>not</u> observed. To a scalar X the operator is applied in the usual way and again might be replaced by the corresponding APL primitive symbol.

Several operations can be combined within one expression. The evaluation is done strictly from right to left, if not differently specified by the use of parentheses.

Table 2 contains a summary of all operators provided by the APL ECONOMETRIC PLANNING LANGUAGE.

<u>Example</u> 6

EXAMPLE 6

```
      0 ST 1 1956 1

      S1 P S2
195401 3446.646 3255.064 3358.905 3405.945

      WL←LOG W1 D W2

      WL
195401 ¯2.2464956 ¯1.2158073 ¯0.45237026 0.019257817

      LL←0 GE LOG W1 D W2

      LL
195401 1 1 1 0

      0 ST 0

      LL←0 GE LOG W1 D W2

      LL
195401 1 1 1 0 0 0 0 0 0 0 0 0 0
```

TABLE 2: Summary of operators defined in the APL ECONOMETRIC PLANNING LANGUAGE

	Operation	System Function Name
Diadic	+	*P*
	-	*M*
	×	*T*
	÷	*D*
	*	*PW*
	<	*L*
	≤	*LE*
	=	*E*
	≥	*GE*
	>	*G*
	≠	*NE*
	∧	*AND*
	∨	*OR*
Monadic	log	*LOG*
	exp	*EXP*
	sin	*SIN*
	cos	*COS*
	tan	*TAN*
	abs value	*ABS*
	cumul value	*CUM*
	~	*NOT*
Spec. Oper.	shifts	*LAG*
	first diff.	*DEL*
	first quot.	*RTO*
	rect.distr.	*RECT*
	norm.distr.	*NORM*
	change period.	*CHANGE*

2.2. SPECIAL OPERATORS

There are some special operators which are frequently used in the formulation of econometric models.

Shift operator: Shifts in time are obtained by the application of the *LAG* function

(*NAME*←) *K* *LAG* *X*

X is the series to be shifted. A positive integer *K* produces a lag of *K* periods, a negative integer *K* a lead of *K* periods. This monadic operator does not observe a time frame setting.

First Differences: The function

(*NAME*←) *DEL* *X*

returns the first difference of *X* which is defined as

X *M* 1 *LAG* *X*

Since this is essentially a dyadic operator (*M*), *DEL* does observe a time frame setting.

First Quotients : The function

(*NAME*←) *RTQ* *X*

returns a result equivalent to

X *D* 1 *LAG* *X*

A time frame setting is observed.

Rectangular Random Numbers: The function

(*NAME*←) *RECT* *K*

evaluates *K* random numbers, uniformly distributed within the unit interval [0,1). If a time series with random number values is to be defined, an appropriate header must be catenated (for *DF* see 1.1.) :

NAME←*HDR* *DF* *RECT* *K*

Normal Random Numbers: *K* random numbers with standard normal distribution N(0,1) are obtained by

(*NAME*←) *NORM* *K*

__Changing the Periodicity of a Time Series:__ If it becomes necessary to generate, e.g., quarterly series from monthly series or vice versa, or in general to change the periodicity of a series, this can be done by means of the *CHANGE* function:

(*NAME*←) *CTR* *CHANGE* *X*

CTR is a two-element control vector. *CTR*[1] denotes the periodicity of the new series (*NAME*). If its periodicity is smaller than that of the old series *X*, several periods of *X* have to be compressed into one new period. By *CTR*[2], one of the following three compression techniques can be selected:

CTR[2] ←	1	average value
	2	cumulative value
	3	last period value.

If the new periodicity is larger, i.e. the series should be expanded, a choice between ten different interpolation techniques can be made:

CTR[2]←	1	stepwise interpolation
	2	linear interpolation
	3	interpol. by compound growth rate
	4	equal parts
	5	sum of digits decreasing
	6	sum of digits increasing
	7	zeros
	8	cumulative equal parts
	9	cum. sum of digits decr.
	10	cum. sum of digits incr.

Techniques 1, 2 and 3 interpolate between two successive values of the old series; techniques 4, 5 and 6 spread the values over the inserted periods; technique 7 sets zeros into the inserted periods; techniques 8, 9 and 10 spread according to 4, 5 and 6, and cumulate.

Either the old (compression) or the new (expansion) periodicity must be an integral multiple of the other periodicity . Otherwise a *CHANGE PER ERR* message is delivered. *CHANGE SPEC ERR* indicates a misspelling of *CTR*[2]. A *CHANGE INT ERR* occurs if interpolation technique 2 or 3 is attempted for a single-period time series *X*.

__Example 7__

EXAMPLE 7

```
      0 ST 1 1956 1

      ES1←(227.2 T W1 D W2) P (.55 T 1 LAG S1) P .12 T DI

      ES1
195501 519.7708 698.05333 958.41953

      RANDOM←4 1970 3 DF NORM 7

      RANDOM
788304 ¯0.5037507 ¯0.42456014 ¯2.0418944 ¯0.30000154 ¯0.56015499
      0.92051797 ¯0.33819378

      T←4 1970 3 DF 15 20 30 50 100 70 40 10

      T11←1 1 CHANGE T
      T12←1 2 CHANGE T
      T13←1 3 CHANGE T

      T21←12 1 CHANGE T
      T22←12 2 CHANGE T
      T23←12 3 CHANGE T
      T24←12 4 CHANGE T
      T25←12 5 CHANGE T
      T26←12 6 CHANGE T
      T27←12 7 CHANGE T
      T28←12 8 CHANGE T
      T29←12 9 CHANGE T
      T210←12 10 CHANGE T
```

```
0 ST 0

10 0 8 2 TABULATE 'T11,T12,T13'
```

	1970	1971	1972
T11	17.50	62.50	25.00
T12	35.00	250.00	50.00
T13	20.00	70.00	10.00

```
0 ST 1 1970 1

LIST←'T21,T22,T23,,T24,T25,T26,,T27,,T28,T29,T210'

10 6 8 2 TABULATE LIST
```

	1970 7	1970 8	1970 9	1970 10	1970 11	1970 12
T21	15.00	15.00	15.00	20.00	20.00	20.00
T22	11.67	13.33	15.00	16.67	18.33	20.00
T23	12.13	13.49	15.00	16.51	18.17	20.00
T24	5.00	5.00	5.00	6.67	6.67	6.67
T25	7.50	5.00	2.50	10.00	6.67	3.33
T26	2.50	5.00	7.50	3.33	6.67	10.00
T27	.00	.00	15.00	.00	.00	20.00
T28	5.00	10.00	15.00	6.67	13.33	20.00
T29	7.50	12.50	15.00	10.00	16.67	20.00
T210	2.50	7.50	15.00	3.33	10.00	20.00

2.3. THE USE OF OPERATORS IN *LIST* ARGUMENTS

Many APL ECONOMETRIC PLANNING LANGUAGE functions such as *TABULATE, PLOT, INTERSECT* or *REGRESS* (see chapter 3) accept a *LIST* of rank one time series as input. It is permissible, where meaningful, to include operations in a *LIST* as demonstrated in the following example.

Example 8

2.4. EQUATIONS AND MODELS

A single equation is made available to the system in the form of an APL-character vector containing a sequence of well-defined operations and an explicit result (dependent variable). As an example consider the first equation of the reference model:

```
EQU1←'S1←(A11 T W1 D W2) P (A12 T 1 LAG S1) P A13 T DI'
```

EQU1 is the user-supplied name of the equation, *S1* the dependent variable. Intermediate results within the same equation are permitted, e.g.

```
EQU1←'S1←(A11 T WR←W1 D W2) P ...'
```

A model is a set of equations. It is presented to the system in the form of a character matrix; each row contains one equation. A function

```
NAME←DMODEL LIST
```

assists the user in shaping this array. *LIST* contains the names of the equations. *NAME* is a user supplied name for the model.

Example 9

(Note that *SALESMODEL* is with the exception of a missing function name the canonical representation of a valid APL function. This fact will be utilized during the solution of the model, which in effect leads to a repeated execution of the corresponding APL function).

EXAMPLE 8

1 1961 1 *ST* 0

10 5 9 3 *TABULATE* '*S*1,*S*2,*S*1 *D* *S*2'

	1961	1962	1963	1964	1965
*S*1	1613.578	1657.487	1731.530	1672.207	1702.119
*S*2	1971.696	1900.684	1808.007	1590.523	1633.202
*S*1 *D* *S*2	.818	.872	.958	1.051	1.042

EXAMPLE 9

```
      EQU1←'S1←(227.2 T W1 D W2) P (.55 T 1 LAG S1) P .12 T DI'

      EQU2←'S2←(¯80.5 T W1 D W2) P (.83 T 1 LAG S2) P .13 T DI'

      EQU3←'W1←.3 P (.34 T S1 D S2) P '
      EQU3←EQU3,'¯.15 T (1 LAG S1) D 1 LAG S2'

      EQU4←'W2←.39 P (¯.29 T S1 D S2) P '
      EQU4←EQU4,'.2 T (1 LAG S1) D 1 LAG S2'

      SALESMODEL1←DMODEL 'EQU1,EQU2,EQU3,EQU4'

      SALESMODEL1
S1←(227.2 T W1 D W2) P (.55 T 1 LAG S1) P .12 T DI
S2←(¯80.5 T W1 D W2) P (.83 T 1 LAG S2) P .13 T DI
W1←.3 P (.34 T S1 D S2) P ¯.15 T (1 LAG S1) D 1 LAG S2
W2←.39 P (¯.29 T S1 D S2) P .2 T (1 LAG S1) D 1 LAG S2
```

2.5. THE USE OF OPERATOR SYMBOLS

The APL ECONOMETRIC PLANNING LANGUAGE includes a translation capability, which permits the user to return to APL operator symbols for most of the mnemonic operator names and also to abbreviate *LAG*, *DEL* and *RTQ* operators. For example a valid alternative for writing *EQU1* in the preceeding paragraph would be:

```
EQU1←'S1←(A11×W1÷W2)+(A12×S1⊂¯1⊃)+A13×DI'
```

However, this form cannot be directly processed by the functions for model solution (chapter 4), but must be translated in advance. This task is performed by the function

```
NAME2←TRANSLATE NAME1
```

NAME1 is the name of an equation or a model containing abbreviated operator symbols. *NAME2* is the name of the translation.

Operations to be immediately executed can also be written in abbreviated form by means of the *DO* function

```
( NAME← ) DO A
```

A is a sequence of operations in character representation, e.g.

```
Z←DO 'X+Y'
```

This is equivalent to

```
Z←X P Y
```

or

```
DO 'Z←X+Y'
```

DO first translates the right argument and then executes it. The last alternative formulation above demonstrates how the *DO* function can be used to solve an equation (of a model), if the right hand side is completely specified. But *DO* does not accept a multi-equation model as input. In this case the solution technique of chapter 4 must be applied.

Table 3 gives a summary of all symbols permitted as abbreviations for mnemonic operator names.

TABLE 3: Valid operator symbols

mnemonic operator name	symbol	mnemonic operator name	symbol
P	+	*G*	>
M	-	*NE*	≠
T	×	*AND*	∧
D	÷	*OR*	∨
PW	*	*NOT*	~
L	<	*K LAG X*	*X*⊂-*K*⊃
LE	≤	*DEL X*	Δ*X*
GE	≥	*RTO X*	⎕*X*

As demonstrated in the following example, the *ASSIGN* function (see 1.1.) can also be used for the decomposition of a model into its component equations. This capability is advantageous if the equations of the final model form are to be stored separately on a file; see the section on DATA FILE ROUTINES.

The last three abbreviations for *LAG*, *DEL* and *RTO* can only be used if the right argument is a name of a series but not a compound expression. Not permitted are, e.g.,

Δ(*X*+*Y*) or (*X*+*Y*)⊂¯1⊃

These should rather be written:

Δ*X*+Δ*Y* or *X*⊂¯1⊃+*Y*⊂¯1⊃

or in long form:

DEL X+*Y* or 1 *LAG X*+*Y*

Example 10

EXAMPLE 10

```
      E1←'S1←(227.4×W1÷W2)+(.55×S1⊂¯1⊃)+.12×DI'

      SALESMODEL2←DMODEL 'E1,EQU2,EQU3,EQU4'

      SALESMODEL2
S1←(227.4×W1÷W2)+(.55×S1⊂¯1⊃)+.12×DI
S2←(¯80.5 T W1 D W2) P (.83 T 1 LAG S2) P .13 T DI
W1←.3 P (.34 T S1 D S2) P ¯.15 T (1 LAG S1) D 1 LAG S2
W2←.39 P (¯.29 T S1 D S2) P .2 T (1 LAG S1) D 1 LAG S2

      SALESMODEL3←TRANSLATE SALESMODEL2

      SALESMODEL3
S1←(227.4 T   W1 D   W2) P   (.55 T   (1 LAG S1)) P   .12 T   DI
S2←(¯80.5 T W1 D W2) P (.83 T 1 LAG S2) P .13 T DI
W1←.3 P (.34 T S1 D S2) P ¯.15 T (1 LAG S1) D 1 LAG S2
W2←.39 P (¯.29 T S1 D S2) P .2 T (1 LAG S1) D 1 LAG S2

      'L1,L2,L3,L4' ASSIGN SALESMODEL3

      L3
W1←.3 P (.34 T S1 D S2) P ¯.15 T (1 LAG S1) D 1 LAG S2

      0 ST 1 1957 1

      DO 'S1+S2'
195401 3446.646 3255.064 3358.905 3405.945 3410.322

      DO L3
195501 0.3357049 0.36697926 0.39319758 0.43409402
```

3. REGRESSION

In section 3, several functions for the statistical estimation of the parameters of an econometric model are described. As mentioned earlier, it is assumed that the model is linear in its parameters. The following estimation techniques are provided.

Ordinary Least Squares
Generalized Least Squares, especially first order autoregressive disturbances
Instrumental Variables Substitution (two-stage least squares)
Principal Components Substitution
Polynomial Distributed Lags (ALMON-Lags)

The principal characteristics of these techniques will be briefly discussed in the subsequent chapters. For a rigorous introduction to the subject the reader is referred to econometric textbooks such as THEIL [8] or SCHNEEWEISS [7]. A concise treatment of the more technical aspects is found in EISNER and PINDYCK [3]

3.1. SIMPLE CORRELATION COEFFICIENTS (THEIL, p. 64)

The pairwise simple correlation coefficients of a *LIST* of time series are calculated by

(*NAME*←) <u>*COR*</u> *LIST*

Only data within the common intersection are included. Of course, this intersection process can be explicitly controlled; see 1.5. Appropriate error messages are printed if the intersection process fails. It should be mentioned again that <u>*COR*</u> and all other functions in this section 3 work only for a *LIST* of <u>rank</u> <u>one</u> series.

<u>Example</u> 11

EXAMPLE 11

```
0 ST 0

C←COR 'S1,S2,W1,W2,DI'

C
 1          ¯0.9771     0.9627    ¯0.9121     0.8461
¯0.9771      1         ¯0.9531     0.9008    ¯0.8972
 0.9627     ¯0.9531     1         ¯0.9516     0.7718
¯0.9121      0.9008    ¯0.9516     1         ¯0.6472
 0.8461     ¯0.8972     0.7718    ¯0.6472     1
```

3.2. ORDINARY LEAST SQUARES (THEIL, pp. 101)

All parameter estimation techniques of the APL ECONOMETRIC PLANNING LANGUAGE must be initiated by calling the function *REGRESS*:

NAME←'Y' REGRESS LIST
WITH:

The function responds with an inquiry *WITH*:, expecting the selection of the desired estimation technique. *REGRESS* continues to respond by means of the keyboard request *WITH*:. In that fashion several techniques can be combined within the same regression. A null carriage return finishes the selection phase.

LIST is the list of independent variables (time series). The presence of the constant term is indicated by a "1", catenated to the list of variables; see example below. *Y* is the dependent variable. The explicit result *NAME* is a character representation of the regression equation including the estimated values for the parameters. In this form the equation can be directly passed on to the construction of the complete model; see 2.4.

REGRESS also prints several statistical measures. The series of disturbances can be obtained by calling the global variable *U* after the regression has concluded. If the disturbances are needed not only immediately but also at a later point a new name should be assigned to them, because the contents of *U* will be destroyed with the next following regression. The covariance matrix of the parameters is contained in the global variable *B*.

Ordinary least squares estimation will be provided if the first *WITH*: is answered by a null carriage return. Only those observations which lie in the intersection of all *LIST* series and the dependent variable *Y* are included in the regression. Again, of course, the intersection process can be controlled explicitly. This is demonstrated in the example where the first three years of the observation data will be excluded from the analysis .

There are two functions to control the output of *REGRESS*. For *K*←0 the command

OUTPUT K

suppresses the print-out of the statistics; for non-zero input *K* it provides a full print-out. For *K*=0, the global variable *B* does not contain the covariance matrix of the parameters but an intermediate result. However the disturbances *U* are still computed correctly.

PRECISION K

with a positive integer $K \le 16$, specifies the significant digits for the parameter values in the explicit result *NAME* of *REGRESS*. The *OUTPUT* and *PRECISION* specifications remain active also for subsequent regressions until they are respecified. A *PRECISION SPEC ERR* occurs for an invalid input *K*.

REGRESS itself might return the following error messages: *REGRESS RANGE ERR* if the intersection of the observation data fails; *REGRESS SING ERR* if the intersection of the series contained in *LIST* has less than full row rank, i.e. some of the series are linearly dependent. Further error messages may be incurred by the selected estimation techniques; see subsequent paragraphs.

Example 12

3.3. GENERALIZED LEAST SQUARES (THEIL, pp. 236)

For ordinary least squares, it is assumed that the disturbances have a scalar covariance matrix in order to yield consistent and efficient estimates:

$C=A\times I$ *A* scalar, *I* unit matrix.

For generalized least squares techniques this assumption is relaxed:

$C=A\times V$ *A* scalar, *V* symmetric positive definite matrix.

The (rare) case, for which *V* is known in advance, can be handled by answering the *WITH* : inquiry with *GLS V*

```
      NAME←'Y' REGRESS LIST
WITH:GLS V
WITH:
```

V is a symmetric, positive definite matrix with as many rows or columns as there are observations. An entry v(s,t) of *V* is, except for an unknown scalar factor, the covariance of the disturbances u(s) and u(t) of periods s and t.

A *GLS DIM ERR* is returned if the dimension of *V* is not properly chosen. *GLS PD ERR* indicates that *V* is not symmetric and positive definite.

```
⍝ EXAMPLE 12

  OUTPUT 1

  PRECISION 6

  1 1956 1 ST 0

      EQU4←'W2' REGRESS '1,S1 D S2,(1 LAG S1) D 1 LAG S2'
WITH:

COEF/VALUE/ST ERR/T-STAT.....

 1   .38506    .04656   8.26945
 2  ¯.29101    .19820  ¯1.46827
 3   .19868    .16595   1.19725

NO OF VARIABLES.........   2.00000
NO OF OBSERVATIONS......  10.00000
SS DUE TO REGRESSION....    .00201
SS DUE TO RESIDUALS.....    .00352
F-STATISTIC.............   1.99692
STANDARD ERROR..........    .02243
R*2 -STATISTIC..........    .36328
R*2 CORRECTED...........    .18136
DURBIN WATSON STATISTIC.   1.13518


      EQU4
W2← ( 0.385065 T 1 ) P ( ¯0.291008 T S1 D S2 ) P ( 0.198681
      T (1 LAG S1) D 1 LAG S2)
```

```
⍝ FOR THE SAKE OF SIMPLICITY IT MAY BE SOMETIMES
⍝ CONVENIENT (BUT NOT NECESSARY) TO DEFINE
⍝ NEW SERIES FOR THE COMPOUND EXPRESSIONS:
0 ST 0
SR←S1 D S2
SL1←1 LAG S1
SL2←1 LAG S2
SRL←SL1 D SL2
WR←W1 D W2

1 1956 1 ST 0


      EQU1←'S1' REGRESS 'WR,SL1,DI'
WITH:

COEF/VALUE/ST ERR/T-STAT.....

 1  227.17691   78.43286    2.89645
 2     .54654     .06132    8.91224
 3     .12025     .03609    3.33226

NO OF VARIABLES.........            3.00000
NO OF OBSERVATIONS......           10.00000
SS DUE TO REGRESSION....    23254099.92031
SS DUE TO RESIDUALS.....       13377.08146
F-STATISTIC.............        4056.15878
STANDARD ERROR..........          43.71512
R*2 -STATISTIC..........            .99943
R*2 CORRECTED...........            .99918
DURBIN WATSON STATISTIC.           2.04822


      EQU1
S1← ( 227.177 T WR ) P ( 0.546535 T SL1) P ( 0.120248 T DI )
```

More important and more frequently used is the specific case of a non-scalar covariance matrix, where the disturbances are generated by a first order autoregressive process (THEIL, pp. 250)

```
U←S P RHO T 1 LAG U
```

with ¯1<*RHO*<1 a scalar constant and *S* a random series with zero expectation and scalar covariance matrix. This case is treated by means of the command A1 *RHO* or A1 ⍳0 if *RHO* is not known in advance:

```
        NAME←'Y' REGRESS LIST
  WITH:A1 RHO    (⍳0)
 (RHO:NN)
  WITH:
```

If *RHO*=⍳0, an estimate for *RHO* is computed (THEIL, p. 254) and printed (*NN*) before the next *WITH* response.

An *A1 VAL ERR* occurs if 1≤|*RHO*. A *REGRESS SING ERR* may be detected before *RHO* is calculated; see 3.2.

It should be noted that after the application of generalized least squares the transformed disturbances *P*+.×*U* are used in the evaluation of the statistics. *P* results from a factorization of the inverse of *V*:

```
((⍉P)+.×P)=⌹V
```

In the case of the first order autoregressive process this corresponds to the use of the disturbances *S* instead of the original *U*. Therefore, after the regression the global variable *U* contains the series *S*.

Example 13

3.4. INSTRUMENTAL VARIABLES SUBSTITUTION (two-stage least squares) (THEIL, p445, pp.451)

One of the basic assumptions for ordinary least squares to yield unbiased, consistent and efficient estimates is, that the right hand side variables in *LIST* are uncorrelated with the disturbance term. This assumption cannot be maintained when the regression equations are part of a simultaneous equations model. For the reference model it can easily be seen that in the first equation *W1 D W2* must be correlated with *U1*, since equations 3 and 4 contain on the right side *S1* and therefore also the disturbance term *U1*.

EXAMPLE 13

```
      0 ST 0

      OUTPUT 0

      PRECISION 4

      ⍝ GENERATION OF A (12,12) RANDOM COVARIANCE MATRIX
      ⍝ WHICH WILL BE USED FOR THE GLS DEMONSTRATION:

      VTEST←12 12⍴RECT 12×12
      VTEST←VTEST+.×⍉VTEST


      GEQU3←'W1' REGRESS '1,SR,SRL'
WITH: GLS VTEST
WITH:


      GEQU3
W1← ( 0.01531 T 1 ) P ( 1.523 T SR ) P ( ¯0.9761 T SRL)


      OUTPUT 1

      PRECISION 6
```

```
      AEQU3←'W1' REGRESS '1,SR,SRL'
WITH: A1 ⍳0
RHO: 0.2297705705
WITH:


COEF/VALUE/ST ERR/T-STAT.....

 1    .07880    .05206  1.51348
 2   1.01861    .30507  3.33890
 3   ¯.60711    .28230 ¯2.15055

NO OF VARIABLES.........      3.00000
NO OF OBSERVATIONS......     12.00000
SS DUE TO REGRESSION....      1.43788
SS DUE TO RESIDUALS.....       .01793
F-STATISTIC.............    240.62091
STANDARD ERROR..........       .04463
R*2 -STATISTIC..........       .98769
R*2 CORRECTED...........       .98358
DURBIN WATSON STATISTIC.      1.65762


      AEQU3
W1← ( 0.0787991 T 1 ) P ( 1.01861 T SR ) P ( ¯0.607108 T SRL
      )
```

In order to remove this deficiency one can apply the technique of instrumental variables substitution. The basic idea behind this technique is to regress all variables in *LIST* which are assumed to be correlated with the disturbance term, first on a set of auxiliary variables (instrumental variables), before the main regression is performed. These auxiliary variables should on the one hand be strongly correlated with the original variables, on the other hand uncorrelated with the disturbance term. The results of these "first stage" regressions replace the original variables in *LIST*. Since the substitutes are linear combinations of the instruments they are also uncorrelated with the disturbance term. The main regression can now take place with the modified *LIST* contents.

It had been shown independently by BASMANN and THEIL, that such a two stage procedure yields asymptotically unbiased, consistent and efficient estimates if all exogenous series of the complete model are used as a set of instruments (THEIL, pp. 497). In that case the procedure had been given the name two-stage least squares technique.

Often, at least for very large models, the set of exogenous variables is quite large and some series are highly collinear, so that a full two-stage least squares procedure would either fail or at best yield inaccurate results. A popular way out is the use of only a subset of the instruments. Here, however, a problem of selecting the best subset arises. This is partly resolved by the use of principal components of the total set. The latter technique will be discussed in the next paragraph.

Instrumental variables substitution is effected by means of the command *INST*

```
      NAME←'Y' REGRESS LIST
WITH: ILIST INST LISTA
WITH:
```

ILIST is the list of instruments, *LISTA* a sublist of *LIST*, containing the series for which the substitution is to be performed. Further *INST* commands can be added if different subsets from *LIST* should be regressed on different instrumental variables:

```
WITH:ILIST1 INST LISTA
WITH:ILIST2 INST LISTB
WITH:
```

The instruments must have the same periodicity and the same number of observations as the original series in *LIST* (after intersection). Otherwise an *INST RANGE ERR* will occur. An *INST SING ERR* is delivered if some of the instruments are linearly dependent. An *INST SPEC ERR* indicates that the right argument of *INST* is not a subset of *LIST*.

The following example demonstrates a full two-stage least squares estimation of equation 2, where all exogenous variables will be used as instruments.

Example 14

3.5. PRINCIPAL COMPONENTS SUSTITUTION (THEIL, pp. 46)

This technique provides the user with a systematic procedure of selecting the right instruments if the complete set of exogenous variables is too large. The basic idea is to transform the complete, normalized set of instruments into a set of as many new series, principal components, which have two properties: They are orthogonal, thus avoiding the problem of collinearity, and they are ordered in a sequence according to their decreasing contribution to the explanation of the original series (in terms of variance decomposition). Finally, as many principal components are selected to be instruments as are needed to exceed a user-specified minimum variance level. The specified variance level should be less than the total variance, because the selection of all principal components is equivalent to the immediate use of all exogenous variables as instruments and no advantage will be gained.

Principal components substitution can be performed by the *PCINST* command

```
      NAME←'Y' REGRESS LIST
WITH:ILIST PCINST LISTA
VARIANCE:K
WITH:
```

As in the previous chapter, *ILIST* is a set of instruments. Here they will, as a first step, be transformed into principal components (using the JACOBI method for calculating eigenvectors). *LISTA* is a subset of *LIST*, for which principal components substitution is to take place. *PCINST* responds with *VARIANCE:*. The answer *K* is either a positive

```
⍝ EXAMPLE 14

      0 ST 0

      IEQU2←'S2' REGRESS 'WR,SL2,DI'
WITH: 'SL1,SL2,1,SRL,DI' INST 'WR'
WITH:

COEF/VALUE/ST ERR/T-STAT.....

 1    32.69428  103.63321      .31548
 2      .86337     .05982    14.43311
 3      .04115     .08363      .49205

NO OF VARIABLES.........             3.00000
NO OF OBSERVATIONS......            12.00000
SS DUE TO REGRESSION....      53964323.76781
SS DUE TO RESIDUALS.....         74063.20131
F-STATISTIC.............          2185.87596
STANDARD ERROR..........            90.71518
R*2 -STATISTIC..........              .99863
R*2 CORRECTED...........              .99817
DURBIN WATSON STATISTIC.             1.72067


      IEQU2
S2← ( 32.6943 T WR ) P ( 0.863371 T SL2) P ( 0.0411482 T DI
      )


      ILIST←'1,SL1,SL2,SRL,DI'
```

real number less than 1, or integral. In the first case, *K* denotes the percentage of total variance to be explained. In the second case, *K* is the number of selected principal components, the variance criterion being ignored. This number should be smaller than the number of series contained in *LIST*, not counting a "1". Again several principal components substitutions can be performed within the same regression; see 3.4.

A *PCINST RANGE ERR* is returned if the instruments in *ILIST* do not have the same periodicity and the same number of observations as the series in *LIST* (after the intersection process). A *PCINST SING ERR* occurs if some of the instruments in *ILIST* are linearly dependent. A *PCINST SPEC ERR* is delivered if *LISTA* is not a subset of *LIST* .

Example 15

3.6. POLYNOMIAL DISTRIBUTED LAGS (ALMON [1])

An equation may incorporate a weighted sum of lags of a series *X*

```
(A0 T X) P (A1 T X[¯1]) P ..... (AK T X[-K])
```

A direct attempt to estimate this term, possibly in combination with other terms in the same equation, may fail because of collinearity or insufficient degrees of freedom. A common approach for resolving this deficiency is to assume a-priori that K+1 parameters are determined by a polynomial of degree less than K. It is also common to impose constraints on the shape of the polynomial, as desired. This usually takes the form of constraining the beginning (head) and/or end (tail) of the polynomial to be zero (see below).

Polynomial distributed lags are performed by a *DIST* command following a *WITH* response. The syntax is

```
        NAME←'Y' REGRESS LIST
    WITH:N DIST 'X'
    WITH:
```

X is the member of *LIST* for which distributed lags are to be estimated. *LIST* contains only the unlagged term; the correct lags are provided automatically at the end; see example below. Within each equation one or more series can be expanded into a weighted sum of lags.

⍝ EXAMPLE 15

```
      PCEQU4←'W2' REGRESS '1,SR,SRL'
WITH: ILIST PCINST 'SR'
VARIANCE: .95
WITH:

COEF/VALUE/ST ERR/T-STAT.....

 1    .50882   .05640  9.02110
 2   ¯.94805   .38712 ¯2.44897
 3    .75369   .35350  2.13208

NO OF VARIABLES.........   2.00000
NO OF OBSERVATIONS......  12.00000
SS DUE TO REGRESSION....    .01501
SS DUE TO RESIDUALS.....    .00928
F-STATISTIC.............   7.27822
STANDARD ERROR..........    .03211
R*2 -STATISTIC..........    .61794
R*2 CORRECTED...........    .53304
DURBIN WATSON STATISTIC.    .95153


      PCEQU4
W2← ( 0.508825 T 1 ) P ( ¯0.948046 T SR ) P ( 0.753688 T SRL
      )
```

The *DIST* command can be called repeatedly during the same regression for different independent variables.

N is a three-element vector. *N*[1] supplies the number of desired terms in the lag structure for *X* (i.e., *K*+1), *N*[2] is the degree of the polynomial, and *N*[3] indicates the constraints imposed: 0 = no constraints, 1 = zero-constraint on head of polynomial, 2 = zero-constraint on tail, 3 = zero-constraint on both head and tail. For the special (common) case where *N*[2 3] = 3 3,these two elements may be omitted and only *N*[1] need be typed.

The original series *X* is used for the construction of the lags, regardless of a previous generalized least squares or an instrumental variables transformation; see 3.7. If *X* has not enough values in front of the current intersection, a reduction of this intersection may occur. A *DIST RANGE ERR* is produced if the number of periods becomes less than or equal to the number of variables. Note, that the effective number of variables is determined by the original contents of *LIST* and the specified degree of the polynomial, taking the specified constraints into account. A *DIST SPEC ERR* indicates a misspelling of one of the arguments of *DIST*.

Example 16

3.7. COMBINATION OF ESTIMATION TECHNIQUES

From a purely technical point of view the user may combine within the same regression different estimation methods in an optional fashion. In the example below, the correction of the disturbance term and an instrumental variables substitution are applied to the same equation. All techniques are executed in the same sequence as they were selected.

However, the chosen sequence affects the statistical properties of the estimates. The theory of correct selection is not yet fully explored. Some aspects are discussed in EISNER and PINDYCK [3]. They suggest the following sequence:

Generalized least squares (*GLS*, *A*1),

instrumental variables substitution (*INST*, *PCINST*),

polynomial distributed lags (*DIST*),

ordinary least squares.

Example 17

```
⍝ EXAMPLE 16


      DLEQU2←'S2' REGRESS 'WR,SL2,DI'
WITH: 4 2 0 DIST 'DI'
WITH:


COEF/VALUE/ST ERR/T-STAT.....

 1  ¯227.48477  216.41694   ¯1.05114
 2     0.80948    0.07789   10.39235
 3    ¯0.34414    0.53270   ¯0.64602
 4    ¯0.29938    0.42354   ¯0.70684
 5     0.08844    0.47402    0.18657
 6     0.81930    0.46762    1.75207

NO OF VARIABLES.........            5.00000
NO OF OBSERVATIONS......           10.00000
SS DUE TO REGRESSION....    38721690.46784
SS DUE TO RESIDUALS.....       35019.72147
F-STATISTIC.............        1105.71098
STANDARD ERROR..........          83.68957
R*2 -STATISTIC..........           0.99910
R*2 CORRECTED...........           0.99819
DURBIN WATSON STATISTIC.           2.64357



      DLEQU2
S2← (¯227.485 T WR ) P ( 0.809484 T SL2 ) P ( ¯0.344138 T DI
 ) P ( ¯0.299375 T 1 LAG DI) P ( 0.0884364 T 2 LAG DI) P ( 0
.819297 T 3 LAG DI)
```

```
⍝ EXAMPLE 17

      1 1956 1 ST 0

      MEQU2←'S2' REGRESS 'WR,SL2,DI'
WITH: A1 ⍳0
RHO: 0.1972740522
WITH: ILIST INST 'WR'
WITH:

COEF/VALUE/ST ERR/T-STAT.....

 1  ¯87.07280  162.74288     ¯.53503
 2     .82908     .07589   10.92410
 3     .12147     .09732    1.24806

NO OF VARIABLES.........            3.00000
NO OF OBSERVATIONS......           10.00000
SS DUE TO REGRESSION....  27630848.94413
SS DUE TO RESIDUALS.....     63664.82282
F-STATISTIC.............      1012.67824
STANDARD ERROR..........        95.36758
R*2 -STATISTIC..........          .99770
R*2 CORRECTED...........          .99672
DURBIN WATSON STATISTIC.         1.65174

      MEQU2
S2← ( ¯87.0728 T WR ) P ( 0.829077 T SL2) P ( 0.121466 T DI
      )
```

```
      MEQU3←'W1' REGRESS '1,SR,SRL'
WITH: A1 ⍳0
RHO: ¯0.1732689385
WITH: ILIST INST 'SR'
WITH:


COEF/VALUE/ST ERR/T-STAT.....

 1    .30444    .05852  5.20197
 2    .33337    .26026  1.28091
 3   ¯.14827    .21698  ¯.68333

NO OF VARIABLES.........        3.00000
NO OF OBSERVATIONS......       10.00000
SS DUE TO REGRESSION....        2.98019
SS DUE TO RESIDUALS.....         .00508
F-STATISTIC.............     1368.46052
STANDARD ERROR..........         .02694
R*2 -STATISTIC..........         .99830
R*2 CORRECTED...........         .99757
DURBIN WATSON STATISTIC.        1.98565
```

4. MODEL SOLUTION

4.1 SOLUTION OF A SIMULTANEOUS EQUATIONS MODEL

A simultaneous equations model which may be nonlinear in its variables is solved in the APL ECONOMETRIC PLANNING LANGUAGE by means of the GAUSS-SEIDEL-technique; see ORTEGA and RHEINBOLDT [6]. The essential idea of this iterative technique is to first normalize the equations of the model in such a way that each unlagged endogenous variable (solution variable) appears on the left hand side of one and only one equation. An initial guess for the first period solution of the left hand side variables being supplied, the model equations are then successively evaluated in order to yield new and improved estimates for the solution. Once a solution for a period is found, the algorithm takes this solution as initial guess for the next period, and the iteration process starts again for that period. The k-th iteration would be (period index omitted):

```
Y1(k)← F1  (Y2(k-1),Y3(k-1),...,Yn(k-1))
Y2(k)← F2(Y1(k), Y3(k-1),...,Yn(k-1))
.
.
.
YN(k)← Fn(Y1(k), Y2(k),...,Yn-1(k))
```

The right hand side of equation n may contain the series Yn only in lagged form, i.e. the term *K* *LAG* Yn with *K*≥1. The criterion for convergence is

EPS *G* *ABS* (Yn(k) *M* Yn(k-1)) *D* Yn(k-1)

for all series Yn. *EPS* is a predefined, sufficiently small positive scalar.

If the convergence process is oscillatory, a relaxation parameter *ALPHA* might provide damping:

Yn(k)←(*ALPHA* <u>*T*</u> Fn(k))<u>*P*</u> (1-*ALPHA*) <u>*T*</u> Yn(k-1)
0<*ALPHA*≤1

where Fn(k) is the right hand side of the unrelaxed process above.

The solution process is initiated by the function <u>*SOLVE*</u>:

LIST <u>*SOLVE*</u> *NAME*

NAME is the name of the model. As mentioned earlier the model must be presented in normalized form. This is not done automatically but has to be considered by the user when defining his equations. It is good practice to specify the model from the very beginning in normalized form, which is always possible by the addition of appropriate definition equations. *LIST* is the list of all exogenous series in the model, excluding lagged endogenous series.

Before <u>*SOLVE*</u> can be requested, several preparatory steps are necessary. First, the <u>simulation</u> <u>horizon</u>, i.e. the periods for which the model should be solved, is to be specified via the setting of a time frame; see 1.5. The horizon may extend over past periods for model validation (ex post simulation) or over future periods for the purpose of forecasting (ex ante simulation).

Secondly, all exogenous series must be available within the active workspace (see e.g. the file routines in chapter 5 as one example to load series). The series must have the proper length in order to fill the model requirements for yielding solutions over the whole simulation horizon; variables appearing in lagged form must include pre-horizon values depending on the lag structure. Also for lagged endogenous series pre-horizon values in proper length must be present.

Initial guesses for the endogenous series within the simulation horizon are computed automatically. Only for the first period, a predefined value is observed. For subsequent periods the solutions of the period before are taken. If no initial value for the first simulation period of an endogenous series is found, the value from the closest period to that is taken (either the last or the first value of that series depending of its time structure). If the series is not defined at all, the square root of 2 is assumed as initial value.

Finally four different parameters are to be set to control the solution process and output

SOLVECON EPS

sets the convergence bound *EPS*. A typical value is *EPS*←.001 (0.1 percent relative error).

SOLVEIT K

defines a limit *K* for the number of iterations per period.

SOLVEREL ALPHA

sets the relaxation parameter *ALPHA*. If it is assumed that the solution process does not oscillate from iteration to iteration then *ALPHA*←1 is appropriate, otherwise a value between zero and one should be specified.

Output is controlled by means of

OUTPUT K

(compare chapter 3.2). For *K*←0 only a convergence message per period including the number of required iterations is printed, otherwise a full convergence diagnostic is produced; see example 18.

The control parameters remain active for repetitive applications of *SOLVE* until they are respecified explicitly. They can also be respecified during execution of *SOLVE* by interruption with the *ATTN* key, execution of *SOLVECON*, *SOLVEIT*, *SOLVEREL* or *OUTPUT*, and return to the interrupted line of *SOLVE*.

After completion of *SOLVE* the solution series are available to the user by the names given in the model; see example 18. If these solutions are to be compared with the observation data for the same variable, their names should be respecified to distinguish between equally named series in the model and on the data file.

If convergence is not achieved within the permitted iteration count, *SOLVE* terminates with a message *ITERATION LIMIT EXCEEDED*. The endogenous series then contain the "solutions" found so far.

A message *MISSING LAGS IN EQU* Yn is delivered if not enough pre-horizon values for a lagged series in that equation are supplied. An *EQU N ERR* occurs if the n'th equation contains a syntax error which prevents the APL execution of the model (internally the model is transformed into an APL function with the name *CMR*). An *EXOG RANGE ERR* indicates that one or several of the exogenous series are not properly specified. An APL length error during execution of *CMR* may result from the omission of an exogenous series in the left argument *LIST* of *SOLVE*. Various intersection or model language errors occur if errors in the model equations are detected during execution.

<u>Example</u> 18

```
      ⍝ EXAMPLE 18
      ⍝ ASSEMBLY OF THE MODEL FROM THE REGRESSION RESULTS
      ⍝ (SECTION 3), ADDING TWO NEW EQUATIONS FOR P1 AND P2

      EQU1
S1←(227.177 T W1 D W2) P (.546535 T 1 LÆ_ S1) P (.120248 T DI)
      MEQU2
S2←(¯87.0728 T W1 D W2) P (.829077 T 1 LAG S2) P (.121466 T DI
)
      MEQU3
W1←(.304443 T 1) P (.333374 T S1 D S2) P (¯.148266 T (1 LAG S1
) D 1 LAG S2)
      EQU4
W2← ( 0.385065 T 1 ) P ( ¯0.291008 T S1 D S2 ) P ( 0.198681 T
(1 LAG S1) D 1 LAG S2)

      EQU5←'P1←S1 D W1'
      EQU6←'P2←S2 D W2'
      EQULIST←'EQU5,EQU1,MEQU3,EQU6,MEQU2,EQU4'

      SALESMODEL←DMODEL EQULIST
      SALESMODEL
P1←S1 D W1

S1←(227.177 T W1 D W2) P (.546535 T 1 LAG S1) P (.120248 T DI)

W1←(.304443 T 1) P (.333374 T S1 D S2) P (¯.148266 T (1 LAG S1
) D 1 LAG S2)
P2←S2 D W2

S2←(¯87.0728 T W1 D W2) P (.829077 T 1 LAG S2) P (.121466 T DI
)
W2← ( 0.385065 T 1 ) P ( ¯0.291008 T S1 D S2 ) P ( 0.198681 T
(1 LAG S1) D 1 LAG S2)

      0 ST 0
      P1←DO EQU5
      P2←DO EQU6
```

```
@ SIMULATION HORIZON

1 1954 1 ST 0

@ I. E. THE SIMULATION COMPRISES EX POST THE WHOLE
@ OBSERVATION PERIOD EXCLUDING THE FIRST YEAR 1953,
@ WHICH IS NEEDED TO FILL THE LAGS IN THE MODEL


@ SPECIFICATION OF CONTROL PARAMETERS

SOLVECON .001
SOLVEIT 100
SOLVEREL 1
OUTPUT 1
```

```
        ⍝ EXECUTION OF SOLUTION ALGORITHM

        'DI' SOLVE SALESMODEL
PER 1  IT 1  MAX DEV 1.805301706 AT VAR W1
NON CONV VARS:
S1,W1,S2,W2
PER 1  IT 2  MAX DEV 0.5354367893 AT VAR P1
NON CONV VARS:
P1,S1,W1,P2,S2,W2
PER 1  IT 3  MAX DEV 0.2504780124 AT VAR P1
NON CONV VARS:
P1,S1,W1,P2,S2
ATTENTION
SOLVE[48]
        OUTPUT 0
        →48
PER 1 1954 1 CONVERGED AT ITERATION 6

PER 1 1955 1 CONVERGED AT ITERATION 6

PER 1 1956 1 CONVERGED AT ITERATION 6

PER 1 1957 1 CONVERGED AT ITERATION 6

PER 1 1958 1 CONVERGED AT ITERATION 6

PER 1 1959 1 CONVERGED AT ITERATION 6

PER 1 1960 1 CONVERGED AT ITERATION 6

PER 1 1961 1 CONVERGED AT ITERATION 7

PER 1 1962 1 CONVERGED AT ITERATION 7

PER 1 1963 1 CONVERGED AT ITERATION 7

PER 1 1964 1 CONVERGED AT ITERATION 7

PER 1 1965 1 CONVERGED AT ITERATION 4
```

```
      ⍝ SOLUTIONS

      S1
195501 711.9592266 1002.738006 1191.858813 1312.992263
1393.424057 1469.837325 1533.138117 1593.946902 1658.992984
1720.965193 1789.765572 1853.13183

      S2
195501 2869.002355 2610.158761 2402.118477 2226.224512
2073.280515 1953.238 1851.484435 1767.282634 1702.933991
1652.393083 1626.219559 1625.935757

      ES1←S1
      ES2←S2
      EW1←W1
      EW2←W2
```

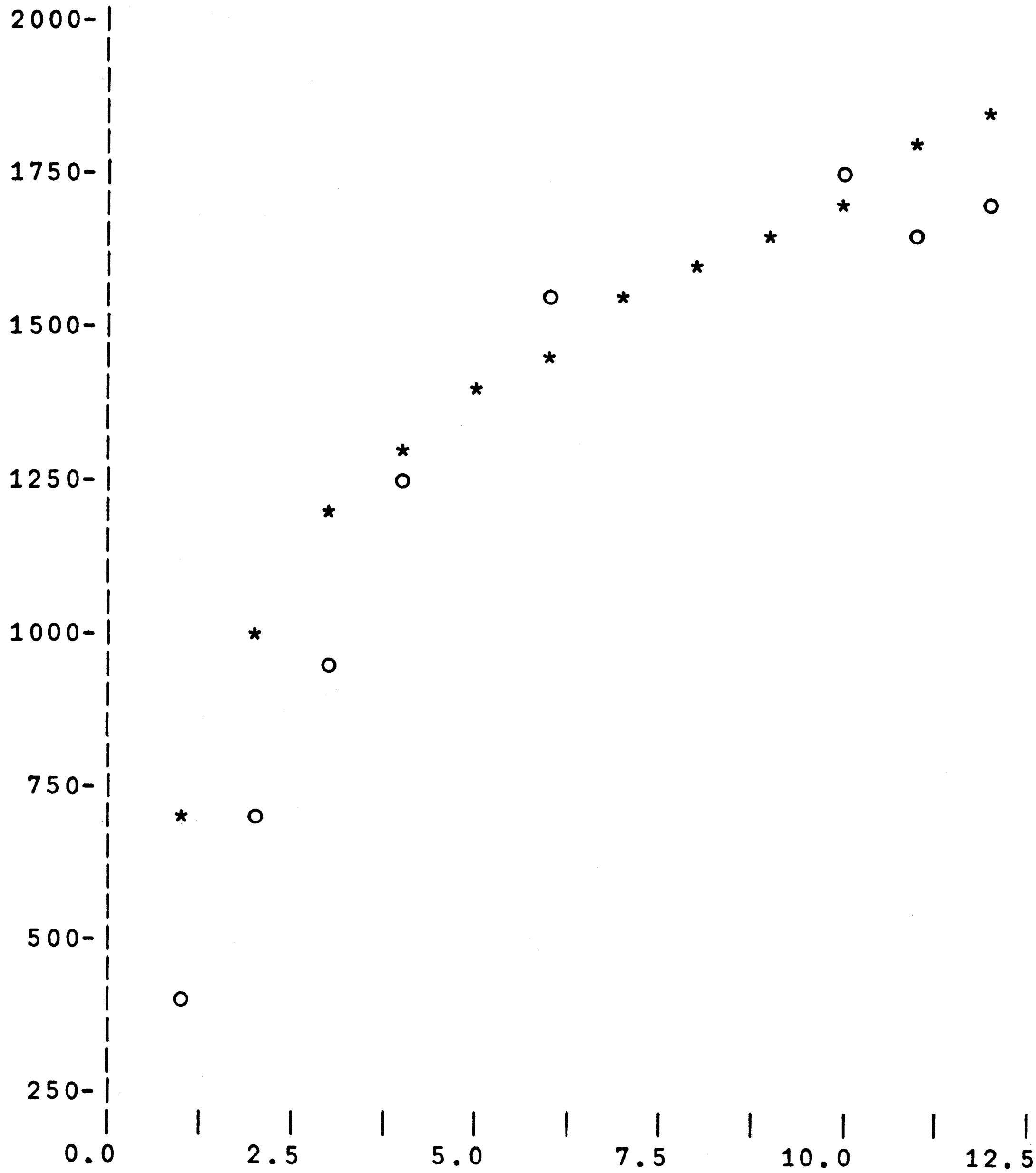
⍝*COMPARISON WITH THE OBSERVED DATA*

```
      )COPY EPLDEMO BASSDATA
SAVED  15.53.13 03/11/76

      LIST←'S1,ES1,,S2,ES2,,W1,EW1,,W2,EW2,,,'

      5 6 9 3 TABULATE LIST
```

	1954	1955	1956	1957	1958	1959
*S*1	398.658	690.081	952.423	1232.111	1414.718	1526.195
*ES*1	711.959	1002.738	1191.859	1312.992	1393.424	1469.837
*S*2	2856.406	2668.824	2453.522	2178.211	2008.075	2006.827
*ES*2	2869.002	2610.159	2402.118	2226.225	2073.281	1953.238
*W*1	0.126	0.236	0.367	0.453	0.451	0.495
*EW*1	0.376	0.396	0.413	0.427	0.441	0.456
*W*2	0.425	0.371	0.360	0.279	0.289	0.287
*EW*2	0.328	0.323	0.317	0.312	0.307	0.300

	1960	1961	1962	1963	1964	1965
*S*1	1571.303	1613.578	1657.487	1731.530	1672.207	1702.119
*ES*1	1533.138	1593.947	1658.993	1720.965	1789.766	1853.132
*S*2	1991.590	1971.696	1900.684	1808.007	1590.523	1633.202
*ES*2	1851.484	1767.283	1702.934	1652.393	1626.220	1625.936
*W*1	0.462	0.455	0.434	0.518	0.506	0.486
*EW*1	0.469	0.482	0.495	0.507	0.517	0.521
*W*2	0.292	0.286	0.283	0.285	0.283	0.317
*EW*2	0.294	0.287	0.281	0.276	0.272	0.272

```
35 50 10 PLOT 'S1,ES1'
```



ABSCISSA = *TIME STARTING FROM* 1954

o = *S*1

* = *ES*1

```
        ⍝ PRODUCING A FORECAST 1966 TO 1968
        ⍝ FORECAST OF EXOGENOUS VARIABLE DI:
        DI←DI,3800 4050 4250
        ⍝ SIMULATION HORIZON
        1 1966 1 ST 1 1968 1
        ⍝ CONTROL PARAMETERS REMAIN UNCHANGED

        'DI' SOLVE SALESMODEL
PER 1 1966 1 CONVERGED AT ITERATION 10
PER 1 1967 1 CONVERGED AT ITERATION 8
PER 1 1968 1 CONVERGED AT ITERATION 7

        10 3 9 3 TABULATE 'S1,S2,,W1,W2,,DI'

                 1966      1967      1968

S1           1821.636  1902.050  1951.880
S2           1649.042  1698.406  1770.613

W1              0.518     0.514     0.506
W2              0.271     0.279     0.287

DI           3800.000  4050.000  4250.000
```

4.2. REORDERING THE SEQUENCE OF THE EQUATIONS OF A MODEL

The APL ECONOMETRIC PLANNING LANGUAGE contains a function *ORDER* which applies VAN DER GIESSEN's algorithm [9] to reorder the sequence of equations of a model. Its aim is an almost recursive model, which contains on the right hand side as few variables as possible that were not already specified on the left hand side of previous equations. Such a model representation has the advantage that the most recent estimates can be used for each iteration during the solution process. Thus, convergence can be accelerated. The reordering algorithm, furthermore, finds a maximum list of endogenous variables which need not be assigned initial estimates because those are automatically calculated during the first iteration.

The reordering is performed by

*NAME*2←*LIST* *ORDER* *NAME*1

*NAME*1 is the name of the unordered, *NAME*2 the name of the ordered model. By means of the *LIST* argument the user can select equations to be excluded from the reordering process. They are placed in the front of the reordered model. *LIST* contains the names of the endogenous series, i.e., the equations with those names as left hand side variables. *LIST*←' ' if all equations should participate in the reordering algorithm.

In addition to the ordered model (as explicit result), *ORDER* returns two lines of information: The sequence of the equations in terms of the left hand side variables and the list of variables for which initial estimates have to be provided before the solution process is entered.

ORDER returns an *ORDER SPEC ERR* message if the names in *LIST* cannot be identified as left hand side variables.

A function

STRUCTURE NAME

displays the causal structure of a model *NAME* in graphic form. The appearance of endogenous variables in the various equations is marked by asterisks. The number of variables with entries above the diagonal is an indication for the degree of recursiveness of the model in the assumed order of equations.

Example 19.1

```
      ∇ EXAMPLE 19.1

      SALESMODEL
P1←S1 D W1

S1←(227.177 T W1 D W2) P (.546535 T 1 LAG S1) P (.120248 T DI)

W1←(.304443 T 1) P (.333374 T S1 D S2) P (¯.148266 T (1 LAG S1
) D 1 LAG S2)
P2←S2 D W2

S2←(¯87.0728 T W1 D W2) P (.829077 T 1 LAG S2) P (.121466 T DI
)
W2← ( 0.385065 T 1 ) P ( ¯0.291008 T S1 D S2 ) P ( 0.198681 T
(1 LAG S1) D 1 LAG S2)


      SALESMODORD←' ' ORDER SALESMODEL
SEQUENCE OF EQUS:
S1,S2,P1,W1,P2,W2

GIVE INIT APPR FOR:
W1,W2


      SALESMODORD
S1←(227.177 T W1 D W2) P (.546535 T 1 LAG S1) P (.120248 T DI)

S2←(¯87.0728 T W1 D W2) P (.829077 T 1 LAG S2) P (.121466 T DI
)
P1←S1 D W1

W1←(.304443 T 1) P (.333374 T S1 D S2) P (¯.148266 T (1 LAG S1
) D 1 LAG S2)
P2←S2 D W2

W2← ( 0.385065 T 1 ) P ( ¯0.291008 T S1 D S2 ) P ( 0.198681 T
(1 LAG S1) D 1 LAG S2)
```

STRUCTURE SALESMODEL

```
      P S W P S W
      1 1 1 2 2 2
     -------------
 P1|  o * *      |P1
 S1|    o *     *|S1
 W1|    * o   *  |W1
 P2|        o * *|P2
 S2|      *   o *|S2
 W2|    *     * o|W2
     -------------
      P S W P S W
      1 1 1 2 2 2
```

```
ROWS    = EQUATIONS
COLUMNS = ENDOG VARIABLES
```

STRUCTURE SALESMODORD

```
      S S P W P W
      1 2 1 1 2 2
     -------------
 S1|  o     *   *|S1
 S2|    o   *   *|S2
 P1|  *   o *    |P1
 W1|  * *   o    |W1
 P2|    *     o *|P2
 W2|  * *       o|W2
     -------------
      S S P W P W
      1 2 1 1 2 2
```

```
ROWS    = EQUATIONS
COLUMNS = ENDOG VARIABLES
```

```
        )COPY EPLDEMO BASSDATA
SAVED   15.53.13 03/11/76

        0 ST 0
        P1←DO EQU5
        P2←DO EQU6


        1 1954 1 ST 1 1965 1

        'DI' SOLVE SALESMODORD
PER 1 1954 1 CONVERGED AT ITERATION 6

PER 1 1955 1 CONVERGED AT ITERATION 5

PER 1 1956 1 CONVERGED AT ITERATION 5

PER 1 1957 1 CONVERGED AT ITERATION 5

PER 1 1958 1 CONVERGED AT ITERATION 5

PER 1 1959 1 CONVERGED AT ITERATION 5

PER 1 1960 1 CONVERGED AT ITERATION 6

PER 1 1961 1 CONVERGED AT ITERATION 6

PER 1 1962 1 CONVERGED AT ITERATION 7

PER 1 1963 1 CONVERGED AT ITERATION 7

PER 1 1964 1 CONVERGED AT ITERATION 7

PER 1 1965 1 CONVERGED AT ITERATION 3
```

4.3. SINGLE EQUATION SOLUTION

For the ex post analysis and validation of a simultaneous equations model it may be desirable to solve each equation separately and independently from the results of other equations. Thereby, all right hand side variables are considered exogenous and must be completely specified before execution. In general, observation data are taken for all these variables rather than model solutions for part of them, as it would be in the simultaneous equations solution. The single equation solution process is straight forward and consists, with the exception of feeding back the dependent variable (see below), of one execution per equation. This is comparable to the application of *DO*(see 2.5).

The function

SSOLVE NAME

performs single equation solution of a model with a specified *NAME* without feedback. Feedback in that context means, that if the left hand side dependent variable is contained in lagged form also on the right hand side of the same equation, the equation result is substituted for the lagged series. Feedback can be applied by calling

SSOLVEF NAME

instead of *SSOLVE*. *NAME* is again the model name.

After execution of either of the two single equation routines the solutions are available by the names given in the model. With *SSOLVE* some of the returned solutions may be shorter because of diadic operations on lagged series in the respective equations.

Example 19.2

```
        ⍝ EXAMPLE 19.2

        )COPY EPLDEMO BASSDATA
 SAVED    9.31.06 01/22/76

        EQU1←'S1←(227.177 T W1 D W2) P (.546535 T 1 LAG S1) P '
        EQU1←EQU1,'(.120248 T DI)'
        EQU2←'S2←(¯87.0728 T W1 D W2) P (.829077 T 1 LAG S2) P '
        EQU2←EQU2,'(.121466 T DI)'
        EQU3←'W1←(.304443 T 1) P (.333374 T S1 D S2) P (¯.148266 '
        EQU3←EQU3,'T 1 LAG S1 D S2)'
        EQU4←'W2←(.385065 T 1) P (¯.291008 T S1 D S2) P (.198681 '
        EQU4←EQU4,'T 1 LAG S1 D S2)'

        SALESMODEL4←DMODEL 'EQU1,EQU2,EQU3,EQU4'

        SALESMODEL4
 S1←(227.177 T W1 D W2) P (.546535 T 1 LAG S1) P (.120248 T DI)
 S2←(¯87.0728 T W1 D W2) P (.829077 T 1 LAG S2) P (.121466 T DI)
 W1←(.304443 T 1) P (.333374 T S1 D S2) P (¯.148266 T 1 LAG S1 D S2)
 W2←(.385065 T 1) P (¯.291008 T S1 D S2) P (.198681 T 1 LAG S1 D S2)


        ⍝ SIMULATION HORIZON

        1 1958 1 ST 1 1965 1

        SSOLVE SALESMODEL4

        S1
 195901 1377.148742 1528.039927 1560.921662 1594.867777 1617.631134
 1715.438371 1770.080409 1702.184749

        W1
 196001 0.4535188862 0.4547082702 0.4602891791 0.4738244781
 0.4944205769 0.5129434986 0.4960040896

        )COPY EPLDEMO BASSDATA
 SAVED    9.31.06 01/22/76


        SSOLVEF SALESMODEL4

        S1
 195901 1377.148742 1507.507012 1550.708023 1583.611901 1601.253612
 1684.704856 1744.488829 1741.689299

        W1
 195901 0.451 0.4535188862 0.4547082702 0.4602891791 0.4738244781
 0.4944205769 0.5129434986 0.4960040896
```

5. DATA FILE ROUTINES

This section describes file usage under APLSV. It also applies to file usage under APL/CMS with the changes noted in 5.4 below.

5.1. SETUP OF A FILE

The APL ECONOMETRIC PLANNING LANGUAGE provides the user with functions for storing, updating and retrieving data into or from a disk storage device. Data can be either time series, cross-sectional series, equations or models. The file functions do not distinguish between form of representation or rank of the data, but consider each data type uniformly as one object.

A file with an optional *NAME* is set up by the function

DDATA 'NAME'

The function asks for a maximum number of objects to be stored and a "maximum number of elements" per object. The second item is used in the allocation of disk space for each object. It should be specified in the following manner:
1. If it is assumed that massive demand for space will arise primarily from long time series, the number of values for the largest series should be supplied.
2. If equations or models are relatively larger, 8 times the number of characters needed to represent the largest object should be supplied. However, it is usually not economical to store a multi-equations model as only one object, but it is preferable to store the component equations separately. After retrieval, the model can be reassembled with the function *DMODEL*; see 2.4.

As with all other functions in this section, error messages refer to TSIO messages. For their encoding the user should consult the APLSV User's Guide (SH20-1460).

5.2. DATA STORAGE, UPDATE AND RETRIEVAL

Once a data file is set up with a specific *NAME*, objects can be stored by means of the function

'NAME' STORE LIST

LIST contains the names of the objects, separated by commas. Objects in *LIST* which were already stored on the file are updated, i.e. the previous contents are replaced by the new.

A message *LISTA NOT SUCCESSFULLY STORED* is printed if the storing of part of the objects, identified in *LISTA*, fails. In addition various TSIO error conditions may occur. If the storage failed due to space allocation problems a new, additional file should be set up. A *NN NOT DEFINED* message indicates that an object *NN* was not found in the workspace.

Retrievals from a file *NAME* are obtained by

'NAME' RETRIEVE LIST

LIST contains the names of the objects to be retrieved.

An *EMPTY DATASET ERR* occurs if retrieval is attempted from a data file which is set up but contains no objects. A *NN NOT FOUND* message is delivered if the object *NN* was not found in the file.

Example 20

5.3. AUXILIARY FILE FUNCTIONS

The function

'NAMENEW' RDATA 'NAMEOLD'

permits the user to change the name of an existing data file.

The function

CDATA 'NAME'

returns the names of all objects stored in *NAME*. An *EMPTY DATASET ERR* indicates a set up file with no loaded object.

To erase a data file *NAME*, one executes

EDATA 'NAME'

5.4 FILE ROUTINES UNDER APL/CMS

File usage under APLSV applies also to APL/CMS, but with the following changes:

The APLSV file routine names have a "C" added under APL/CMS: *DDATAC*, *STOREC*, *RETRIEVEC*, *RDATAC*, *CDATAC*, *AND EDATAC*.

In *DDATAC* the user is not prompted for size of space needed.

The error messages are generated by AP110, and are listed in the APL/CMS User's Manual (SC20-1846).

```
      ⍝ EXAMPLE 20

      DDATA 'SLSMODEL'
MAX NO OF  OBJECTS: 20
MAX NO OF ELEMENTS: 25

      ⍝ (THE NAME OF THE FILE SHOULD NOT BE LONGER
      ⍝ THAN 8 CHARACTERS!)


      LIST1←'E1,E2,E3,E4,E5,E6'

      LIST1 ASSIGN SALESMODORD

      E1
S1←(227.177 T W1 D W2) P (.546535 T 1 LAG S1) P .120248 T DI

      LIST←LIST1,',S1,S2,W1,W2'
      LIST
E1,E2,E3,E4,E5,E6,S1,S2,W1,W2

      'SLSMODEL' STORE LIST

      )ERASE E1 S1

      'SLSMODEL' RETRIEVE 'S1,E1'

      E1
S1←(227.177 T W1 D W2) P (.546535 T 1 LAG S1) P .120248 T DI

      EDATA 'SLSMODEL'
```

6. ADDENDUM

The addendum contains four additions to the system , with the functional objectives:

1. retranslation of system operators into APL operators,

2. output of tables with text,

3. updating of time series,

4. adjustment of periodicities for time series.

6.1. RETRANSLATION: *RETRANS*

The function translates operators from the system representation to the APL representation. Its argument can be the character representation of either one equation or of a model with system operator symbols. See example 21.

The retranslation function calls a function *VOCABULARY* which contains a list of the vocabulary currently used. This vocabulary can be altered or extended subject to the following rules:

> The APL operators on the right hand side of the vocabulary must not be longer than the system operators.
>
> In case they are shorter, (e.g., *PW* might be translated into *), the APL character ¨ must be used as fill-character to make the lengths of the two operators equal.

```
⍝ EXAMPLE 21

      E1←'A←B P C PW (A D B) M (1 LAG A)'

      E2←'C←NOT A AND B'

      MOD←DMODEL 'E1,E2'

      E1
A←B P C PW (A D B) M (1 LAG A)

      E2
C←NOT A AND B

      MOD
A←B P C PW (A D B) M (1 LAG A)
C←NOT A AND B


      RETRANS E1
A←B + C * (A ÷ B) - (1 LAG A)

      RETRANS E2
C←~ A ∧ B

      RETRANS MOD
A←B + C * (A ÷ B) - (1 LAG A)
C←~ A ∧ B
```

6.2. OUTPUT OF TABLES WITH TEXT: *TABULTEXT*

This function is a minor extension of *TABULATE*. Using the same arguments as *TABULATE* it allows the use of a header and the insertion of text in place of variables for which row information is to be printed. The information is passed by means of global variables defined by the user.

The desired header is placed into the variable: *TABHEAD*.

Row text in place of a variable name is placed into a variable *TEXT*(name). E.g, text for the variable S1 should be put into the variable *TEXTS*1.

When the variables *TABHEAD* and *TEXT*... do not exist, the output of *TABULTEXT* is the same as that for *TABULATE*.

See example 22 on the next page.

6.3. UPDATING OF TIME SERIES: *UPDATE*

It may be desirable to alter values of time series or to insert new values. *UPDATE* permits:

- updating of coefficients within the range of the time series,
- extension of the old time series.

The use of *UPDATE* will be made clear by example 23.

EXAMPLE 22

```
TABHEAD←'SALES PER CAPITA OF FILTER AND NONFILTER CIG'
TABHEAD←TABHEAD,'ARETTES BETWEEN 1953 AND 1965'

TEXTS1←'SALES BRAND 1'
TEXTS2←'      BRAND 2'

      S1
195401 250.328 398.658 690.081 952.423 1232.111 1414.718
      1526.195 1571.303 1613.578 1657.487 1731.53
      1672.207 1702.119

      S2
195401 3196.318 2856.406 2668.824 2453.522 2178.211
      2008.075 2006.827 1991.59 1971.696 1900.684
      1808.007 1590.523 1633.202

      15 0 9 1 TABULTEXT 'S1,S2'
```

SALES PER CAPITA OF FILTER AND NONFILTER CIGARETTES BETWEEN 1953 *AND* 1965

	1953	1954	1955	1956	1957
SALES BRAND 1	250.3	398.7	690.1	952.4	1232.1
BRAND 2	3196.3	2856.4	2668.8	2453.5	2178.2

	1958	1959	1960	1961	1962
SALES BRAND 1	1414.7	1526.2	1571.3	1613.6	1657.5
BRAND 2	2008.1	2006.8	1991.6	1971.7	1900.7

	1963	1964	1965
SALES BRAND 1	1731.5	1672.2	1702.1
BRAND 2	1808.0	1590.5	1633.2

```
      ⍝ EXAMPLE 23

      TT←4 1950 2 DF ⍳8

      TT
780204 1 2 3 4 5 6 7 8

      DISPLAY 'TT'

VARIABLE TT

PERIODICITY = 4
ORIGIN = 1950 2
NO OF ENTRIES = 8

    TIME      VALUE        TIME      VALUE

  1950    2   1.00000   1951    2   5.00000
  1950    3   2.00000   1951    3   6.00000
  1950    4   3.00000   1951    4   7.00000
  1951    1   4.00000   1952    1   8.00000

      TTT←UPDATE TT
FIRST PERIOD TO BE UPDATED
⎕:
      4 1951 2
NEW VALUE(S)
⎕:
      1.5 2.5
RESPECTIVE OLD VALUE(S):
5 6

      DISPLAY 'TTT'

VARIABLE TTT

PERIODICITY = 4
ORIGIN = 1950 2
NO OF ENTRIES = 8

    TIME      VALUE        TIME      VALUE

  1950    2   1.00000   1951    2   1.50000
  1950    3   2.00000   1951    3   2.50000
  1950    4   3.00000   1951    4   7.00000
  1951    1   4.00000   1952    1   8.00000
```

```
      TTT←UPDATE TT
FIRST PERIOD TO BE UPDATED
⎕:
      4 1949 2
UPDATING POSITION BEFORE BEGINNING OF SERIES.
NEW VALUE(S)
⎕:
      1.5 2.5 3.5 4.5
RESPECTIVE OLD VALUE(S):
0 0 0 0
      DISPLAY 'TTT'


VARIABLE TTT

PERIODICITY = 4
ORIGIN = 1949 2
NO OF ENTRIES = 12

    TIME      VALUE        TIME      VALUE

  1949   2  1.50000  1950   4  3.00000
  1949   3  2.50000  1951   1  4.00000
  1949   4  3.50000  1951   2  5.00000
  1950   1  4.50000  1951   3  6.00000
  1950   2  1.00000  1951   4  7.00000
  1950   3  2.00000  1952   1  8.00000

      TTT←UPDATE TT
FIRST PERIOD TO BE UPDATED
⎕:
      4 1949 4
UPDATING POSITION BEFORE BEGINNING OF SERIES.
NEW VALUE(S)
⎕:
      9 8 7 6
RESPECTIVE OLD VALUE(S):
0 0 1 2
      DISPLAY 'TTT'


VARIABLE TTT

PERIODICITY = 4
ORIGIN = 1949 4
NO OF ENTRIES = 10

    TIME      VALUE        TIME      VALUE

  1949   4  9.00000  1951   1  4.00000
  1950   1  8.00000  1951   2  5.00000
  1950   2  7.00000  1951   3  6.00000
  1950   3  6.00000  1951   4  7.00000
  1950   4  3.00000  1952   1  8.00000
```

```
      TTT←UPDATE TT
FIRST PERIOD TO BE UPDATED
⎕:
      4 1952 2
NEW VALUE(S)
⎕:
      1.6 9.3
RESPECTIVE OLD VALUE(S):
0 0

      DISPLAY 'TTT'

VARIABLE TTT

PERIODICITY = 4
ORIGIN = 1950 2
NO OF ENTRIES = 10

    TIME      VALUE        TIME     VALUE

  1950   2  1.00000  1951    3  6.00000
  1950   3  2.00000  1951    4  7.00000
  1950   4  3.00000  1952    1  8.00000
  1951   1  4.00000  1952    2  1.60000
  1951   2  5.00000  1952    3  9.30000

      TTT←UPDATE TT
FIRST PERIOD TO BE UPDATED
⎕:
      4 1952 4
GAP BETWEEN FIRST PERIOD TO BE UPDATED AND LAST PERIOD OF THE OLD
      TIMESERIES.
INPUT FOR THESE PERIODS IS ASSUMED TO BE ZERO.
NEW VALUE(S)
⎕:
      11.9 96.3
RESPECTIVE OLD VALUE(S):
0 0

      DISPLAY 'TTT'

VARIABLE TTT

PERIODICITY = 4
ORIGIN = 1950 2
NO OF ENTRIES = 12

    TIME      VALUE        TIME      VALUE

  1950   2  1.00000  1951    4   7.00000
  1950   3  2.00000  1952    1   8.00000
  1950   4  3.00000  1952    2    .00000
  1951   1  4.00000  1952    3    .00000
  1951   2  5.00000  1952    4  11.90000
  1951   3  6.00000  1953    1  96.30000
```

6.4. ADJUSTMENT OF PERIODICITY: *PREP*

The right argument of *PREP* specifies the time series to be altered, the left argument gives the new periodicity. The adjustment may either correspond to a compression or to an expansion (interpolation) of the time series. For each time series in the right argument, then, one has to specify the desired method of compression or interpolation.

We associate a variable *PERCHAN*(name) with the time series (name). If this variable exists, it is used during execution of *PREP*, if not it is generated through a question at the terminal. The first component of *PERCHAN*(name) specifies a method for compression, the second a method for interpolation. It will always be clear from the context which one is to be used.

Consult the description of *CHANGE* for possible error messages. Various possibilities are demonstrated in example 24.

EXAMPLE 24

```
      T1←4 1950 3 DF 1 2 3 4 5 6 7 8 9
      T2←12 1960 7 DF ι30
      T2
2352712 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
      19 20 21 22 23 24 25 26 27 28 29 30

      2 PREP 'T1,T2'
STATE COMPRESSION AND INTERPOLATION METHOD FOR T1
3 2
PERCHANT1: 3 2
STATE COMPRESSION AND INTERPOLATION METHOD FOR T2
2 1
PERCHANT2: 2 1
      T1
390202 2 4 6 8 9
      T2
392202 21 57 93 129 165
      EC T1[1]
2 1950 2
      EC T2[1]
2 1960 2

      12 PREP 'T1,T2'
PERCHANT1: 3 2
PERCHANT2: 2 1
      T1
2340712 0.3333333333 0.6666666667 1 1.333333333 1.666666667
      2 2.333333333 2.666666667 3 3.333333333 3.666666667
      4 4.333333333 4.666666667 5 5.333333333 5.666666667
      6 6.333333333 6.666666667 7 7.333333333 7.666666667
      8 8.166666667 8.333333333 8.5 8.666666667 8.833333333
      9
      T2
2352712 21 21 21 21 21 21 57 57 57 57 57 57 93 93
      93 93 93 93 129 129 129 129 129 129 165 165
      165 165 165 165
      EC T1[1]
12 1950 7
      EC T2[1]
12 1960 7

      52 PREP 'T1,T2'
PERCHANT1: 3 2
CHANGE PER ERR
WITH SERIES: T1
PERCHANT2: 2 1
CHANGE PER ERR
WITH SERIES: T2
```

REFERENCES

1. Almon, S., The Distributed Lag between Capital Appropriations and Expenditures, Econometrica, 33(1965), p. 178-196

2. Bass, F.M., A Simultaneous Equations Regression Study of Advertising and Sales of Cigarettes, J. of Marketing Research, 6(1969), p. 291-300

3. Eisner, M. and R. Pindyck, A Generalized Approach to Estimation as Implemented in the TROLL/1 System, Annals of Economic and Social Measurement, 2(1973), p.29-52

4. Falkoff, A.D. and K.E. Iverson, APL Shared Variables, APLSV User's Guide, IBM Corporation, 1973, Manual SH20-1460

5. Graphs and Histograms in APL/360-370, IBM Corporation, Manual SB21-0415, 1972

6. Ortega, J. and W. Rheinboldt, Iterative Solution of Nonlinear Equations in Several Variables, Academic Press, New York, 1970

7. Schneeweiss, H., Oekonometrie, Physica Verlag, Wuerzburg, 1971

8. Theil, H., Principles of Econometrics, J. Wiley, New York, 1971

9. Van der Giessen, A.A., Solving Non-linear Systems by Computer; a New Method, Statistica Neerlandica, 24(1970), p.1-10

10. Schober, F., Eine interaktive oekonometrische Modellsprache in APL, in Hansen, H.R., Computergestuetzte Marketing-Planung, Verlag Moderne Industrie, Muenchen, 1974

11. Schober, F., Econometric Modeling in APL, Proceedings, COMPSTAT Symposium, Vienna, Sept. 1974

12. Schober, F. and K. Spielberg, Application Design in APL: The APL Econometric Planning Language, Working Paper, to be submitted to IBM Systems Journal, 1976

SH20-1620-1

APL Econometric Planning Language for System/370 OS/VS1 or OS/VS2 or
VM/370 CMS Program Description/Operations Manual Printed in U.S.A. SH20-1620-1

IBM

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, New York 10604
(U.S.A. only)

IBM World Trade Corporation
360 Hamilton Avenue, White Plains, New York 10601
(International)

READER'S COMMENT FORM

APL Econometric Planning Language for
System/370 OS/VS1 or OS/VS2 or VM/370 CMS
Program Description/Operations Manual

SH20-1620-1

Please comment on the usefulness and readability of this publication, suggest additions and deletions, and list specific errors and omissions (give page numbers). All comments and suggestions become the property of IBM. If you wish a reply, be sure to include your name and address.

COMMENTS

Fold

Fold

Fold

Fold

- Thank you for your cooperation. No postage necessary if mailed in the U.S.A.
FOLD ON TWO LINES, STAPLE AND MAIL.

SH20-1620-1

Your comments, please . . .

This manual is part of a library that serves as a reference source for systems analysts, programmers, and operators of IBM systems. Your comments on the other side of this form will be carefully reviewed by the persons responsible for writing and publishing this material. All comments and suggestions become the property of IBM.

Fold | Fold

First Class
Permit 40
Armonk
New York

Business Reply Mail

No postage stamp necessay if mailed in the U.S.A.

Postage will be paid by:

International Business Machines Corporation
1133 Westchester Avenue
White Plains, New York 10604

Att: Technical Publications/Industry – Dept. 825

Fold | Fold

IBM

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, New York 10604
(U.S.A. only)

IBM World Trade Corporation
360 Hamilton Avenue, White Plains, New York 10601
(International)

APL Econometric Planning La... for System/370 OS/VS1 or OS/VS2 or VM/370 CMS Program Descri.../Operations Manual Printed in U.S.A. SH20-1620-1